공학박사학위논문

분산된 로터로 구동되는 비행 스켈레톤 시스템의 디자인, 상태추정 및 제어

Design, Estimation and Control of an Aerial Skeleton System with Distributed Rotor Actuation

2020 년 2 월

서울대학교 대학원

기계항공공학부

박 상 율

분산된 로터로 구동되는 비행 스켈레톤 시스템의 디자인, 상태추정 및 제어

Design, Estimation and Control of an Aerial Skeleton System with Distributed Rotor Actuation

지도교수 이 동 준

이 논문을 공학박사 학위논문으로 제출함

2019 년 10 월

서울대학교 대학원

기계항공공학부

박 상 율

박상율의 공학박사 학위논문을 인준함

2019 년 12 월

위 원 장 : 박 종 우 (인)

부위원장 : 이 동 준 (인)

위 원 : 조 규 진 (인)

위 원 : 이 제 희 (인)

위 원 : 양 기 훈 (인)

Abstract

In this thesis, we present key theoretical components for realizing flying aerial skeleton system called LASDRA (large-size aerial skeleton with distributed rotor actuation). Aerial skeletons are articulated aerial robots actuated by distributed rotors including both ground connected type and flying type. These systems have recently attracted interest and are being actively researched in several research groups, with the expectation of applying those for aerial manipulation in distant/narrow places, or for the performance with entertaining purpose such as drone shows. Among the aerial skeleton systems, LASDRA system, proposed by our group has some significant advantages over the other skeleton systems that it is capable of free SE(3) motion by omni-directional wrench generation of each link, and also the system can be operated with wide range of configuration because of the 3DOF (degrees of freedom) inter-link rotation enabled by cable connection among the link modules.

To realize this LASDRA system, following three components are crucial: 1) a link module that can produce omni-directional force and torque and enough feasible wrench space; 2) pose and posture estimation algorithm for an articulated system with high degrees of freedom; and 3) a motion generation framework that can provide seemingly natural motion while being able to generate desired motion (e.g., linear and angular velocity) for the entire body. The main contributions of this thesis is theoretically developing these three components, and verifying these through outdoor flight experiment with a real LASDRA system. First of all, a link module for the LASDRA system is designed with proposed constrained optimization problem, maximizing the guaranteed feasible force and torque for any direction while also incorporating some constraints (e.g., avoiding inter-rotor air-flow interference) to directly obtain feasible solution. Also, an issue of ESC-induced (electronic speed control) singularity is first introduced in the literature which is inevitably caused by bi-directional thrust generation with sensorless actuators, and handled with a novel control allocation called selective mapping. Then for the state estimation of

the entire LASDRA system, constrained Kalman filter based estimation algorithm is proposed that can provide estimation result satisfying kinematic constraint of the system, also along with a semi-distributed version of the algorithm to endow with system scalability. Lastly, CPG-based motion generation framework is presented that can generate natural biomimetic motion, and by exploiting the inverse CPG model obtained with machine learning method, it becomes possible to generate certain desired motion while still making CPG generated natural motion.

Keywords: aerial skeleton, design optimization, ESC-induced singularity, scalability, constrained Kalman filter, central pattern generator, machine learning
Student Number: 2013-20672

Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation and Background

Large-size robots are systems that people have always dreamed of, as we can frequently see those in various sci-fi movies, animations, or rarely as real robots in [1, 2]. However, it is still challenging to realize those kind of systems, because of the scalability issue with conventional actuators including electric motors and hydraulic actuators. Namely, as the length or the number of articulated links of a robot increases, the mass of links are accumulated as load along the system requires the size of the base actuator to grow exponentially to endure and control all the system, as addressed in [3].

Meanwhile, in recent years, a novel platform named as "aerial skeleton" has been an interest of several research groups, such as [3, 4, 5, 6, 7, 8]. Aerial skeleton refers to an aerial robot with articulated structure which is mainly actuated by the thrust of distributed rotors (may contain additional actuators to rotate the rotors with respect to the link and/or to directly produce inter-link relative motion), and this platform is expected to overcome the aforementioned scalability issue by compensating load of each link with thrusters, while being actuated by pushing the surrounding air (or also dubbed as external actuation [3]). In [5], the Hiryu-I

system is constructed with parallel link mechanism and 2-DOF (degree-of-freedom) joints actuated by thrusters attached on each link, in [6, 7], a two dimensional multi-link aerial robot is propose with servo motors for each 1-DOF joint, while modified design is proposed in [7] to guarantee full actuation using tilted rotors, and in [8], the DRAGON (dual-rotor multi-link robot with ability of multi-DOF aerial transformation) system is developed utilizing servo-motors to generate both rotor-link and link-link motions. With the size-scalable property, these systems are envisioned to realize such new applications as mechanical operations at high-rise building or in a narrow/long space; or articulated flying characters in amusement parks. However, the aforementioned systems have some limitations that the degree of freedom of each joint in [6, 7] is limited to one limiting the possible range of motion or configuration, and the systems in [5, 8] have singularity issue that those cannot be controlled at upright configuration because of the under-actuation at each link.

In this context, a novel aerial robot called LASDRA [3], [4] (Large-size Aerial Skeleton with Distributed Rotor Actuation) is developed in our research group, which is constructed with link modules that can generate omni-directional wrench, enabling free SE(3)-motion of each link. Also, each link of LASDRA is connected with its neighbouring links via cable, so that wide range of posture becomes available with 3 DOF inter-link rotation. We developed two types of LASDRA system, one is the ground-connected type called o-LASDRA (operation-LASDRA) system [3], and the other is the free flying type called f-LASDRA (flying LASDRA) system [4] . The former type can be exploited for manipulation tasks at places where mobile robots or conventional manipulators cannot reach, with almost unlimited operating time enabled by continuous supply of power, and the latter type system is expected to be successfully applied for entertainment purpose such as drone shows [9]. It is notable here that, drone shows are performing with individual dots while the f-LASDRA system can perform with connected lines using its own skeleton structure, which makes it look much more natural and gratifying.

1.2 Research Problems and Approach

In this thesis, we first focus on the development of the link module called ODAR (omni-directional aerial robot) that can be used for constructing both types of LASDRA system. To freely operate the LASDRA system in three dimensional space while always compensating for the load of each link, we design each link of the system or ODAR to be able to generate omni-directional wrench. Since the omni-directional wrench generation can be achieved by attaching rotors non-colinearly with the sacrifice of the force generation efficiency, it requires more thrust to fly compare to typical multi-rotor platforms with same mass, and it becomes crucial to maximize the wrench space with given number of rotors. Here, we do not adapt the concept of adding servo motors to tilt the rotors to guarantee full actuation [10, 11, 12, 13], since it reduces already tight payload with additional actuators, and it cannot generate desired wrench immediately due to the limited bandwidth of servo motors. To resolve this issue, we formulated constrained optimization problem to decide the attached position and direction of the rotors maximizing the guaranteed minimum control force and torque for any direction. Also, we consider constraints including volume constraint for the position of rotors, self weight compensation, and avoidance of inter-rotor air flow interference for the optimization, so that feasible and directly implementable solution can be obtained from the optimization.

Although with the design optimized ODAR system, there exists an hardware issue when we truly operate this system omni-directionally. From the experiment of rotating motion in pitch direction, we found out that the system shows shaky motion and even results in fall down. This phenomenon appears when some of rotors are commanded to change their rotating direction, and we found that reason of the shaky motion is the hesitating behaviour of the motor when it changes the rotating direction. This behaviour is inevitable with common sensorless motors used for drones since the motor relies on back-EMF (electromotive force) signal for commutation and the motor lacks this signal at low RPM, so the motor needs to stop

first and reaccelerate to the other direction to change the rotating direction, causing delayed thrust output. In this thesis, we name this phenomenon as "ESC-induced (electronic speed control) singularity", and we propose a novel control allocation method to overcome this issue, the idea of which is not using the rotor that originally generate near-zero thrust, exploiting redundancy of actuation given by 8 rotors.

Now, let us describe some issues of operating the f-LASDRA system in the outdoor environment. For the system modularity and scalability, the f-LASDRA system is equipped with IMU and GPS module for each link, and distributed impedance control is applied for all the links, so that pose sensing and control can be done independently. Then, there exists issue on position sensing of each link which is important for the distributed impedance control. When we use IMU attitude estimate and forward kinematics for a position estimation of each link, attitude error and noise are accumulated throughout the skeleton making the estimate unreliable. Also, the GPS sensor is well known for its poor accuracy and slow rate, so if we only rely on the IMU/GPS based pose estimation on each link, the estimated result would not satisfy the kinematic constraint given by 3 DOF joints between the links. This will cause excessive internal force for joints, and it can lead to a saturation of rotor thrust or also a self-collision and fall down. Therefore, it is critical to have pose estimation result of each link that is coherent to the kinematic structure of the system. To deal with this issue, we propose a novel estimation framework exploiting individual IMU/GPS sensors and kinematic constraints of the system together, which is based on SCKF (smoothly constrained Kalman filter) [14].

Considering that the major application of the f-LASDRA system would be a performance with entertaining purpose similar to drone shows, generating natural motion is another important goal for the system. To achieve this goal, we first define the motion naturalness of the f-LASDRA system as a combination of natural biomimetic shape motion and the entire body motion that matches the physics that people expect. We denote the shape motion and the body motion here as a link motion seen from the body frame, and the motion of body frame seen from the

inertial frame where the body frame is the representative frame of the whole system. The natural shape motion is then provided from CPG (central pattern generator) model [15] which is formulated to generate the biomimetic motion referring to a biological experiment data. Here, we adapted CPG model for the motion generation as it can provide coordinated shape motion with high DOF only with few parameters, and continuous/differentiable rhythmic motion can be obtained even with parameter transition. Then, the motion generated from CPG model is simulated with the expected physics and the resultant motion becomes the final target pose for each link of the system. Lastly, we learn the inverse map of CPG parameter to generated motion using MLP (multi layer perceptron) and autoencoder, so that we can directly generate some target motion while incorporating motion naturalness from CPG-based motion generation.

1.3 Preview of Contributions

The main contribution of this thesis can be summarized as developing key components required for realizing outdoor flying LASDRA system, and those key components include: 1) developing a link module called ODAR that can produce omni-directional force and torque; 2) proposing a constrained Kalman filter based pose estimation algorithm for an articulated system with high degrees of freedom; and 3) proposing a CPG based motion generation framework that can provide natural biomimetic motion.

There have been abundant research about designing fully actuated aerial vehicles as in [16, 17, 18, 19, 20], but there are no other works except for this thesis that optimize both location and direction of rotors while properly considering constraints for the feasibility of a solution. In addition, in this thesis, the problem of ESC-induced singularity is first introduced in the literature, and handled with a novel control allocation called selective mapping. The CPG based motion generation framework along with the inverse CPG model from machine learning and

the parameter adaptation is also a new approach that has never been presented by others.

The contributions presented in this thesis are not limited to be used for flying LASDRA system, but can be extended to other robotic systems. First of all, the design optimization framework can be applied for other thrust propelled systems such as AUVs (autonomous underwater vehicle). The constrained Kalman filter based state estimation algorithm can be also applied for articulated system with high degrees of freedom including humanoid, hyper-redundant manipulators, and so on. Furthremore, CPG based motion generation framework can be extended to other robotic systems which is addressed and verified with simulation in this thesis.

The outline of this thesis is as follows. In chapter 2, we describe the design and control problems of an omni-directional aerial robot, which is exploited as a single link for constructing the LASDRA system. Then in chapter 3, pose and posture estimation framework for LASDRA system is presented enforcing kinematic coherency. Motion generation method for motion naturalness based on CPG is described in chapter 4, and in chapter 5, experiment results of outdoor flying LASDRA system are presented. Lastly in chapter 6, we summarize the contributions of this thesis, and conclude with description of some possible future works.

Chapter 2

Omni-Directional Aerial Robot

2.1 Introduction

Multi-rotor unmanned aerial vehicles (UAVs) or simply drones have received booming interests from the research community and the general public alike due to their capacity/promise to extend our sensory and manipulation ability to the 3D-space without being bound to the ground. This flourishing field of drones is enabled by the recent advancement and maturation of many background technologies, including material/manufacturing (e.g., carbon fiber, magnesium alloy), sensors (e.g., MEMs (micro-electromechanical systems) IMU (inertial measurement unit), sonar, cameras, etc.), actuators (e.g., BLDC (brush-less direct-current) motors), onboard computing and communication, and algorithms (e.g., sensor fusion, localization, control, image processing, etc.), to name just few.

The most successful applications of the drones so far are mostly "seeing" applications, including aerial photography, geo-surveying, traffic monitoring, etc. However, to truly extend the usefulness of the drones to the 3D-space, it is necessary to endow them with the ability of aerial manipulation, and for that, drone-manipulator systems (i.e., drone with multi-degree-of-freedom (DOF) robotic arm) are most intensively investigated (e.g., [21, 22, 23, 24, 25]). This drone-manipulator system,

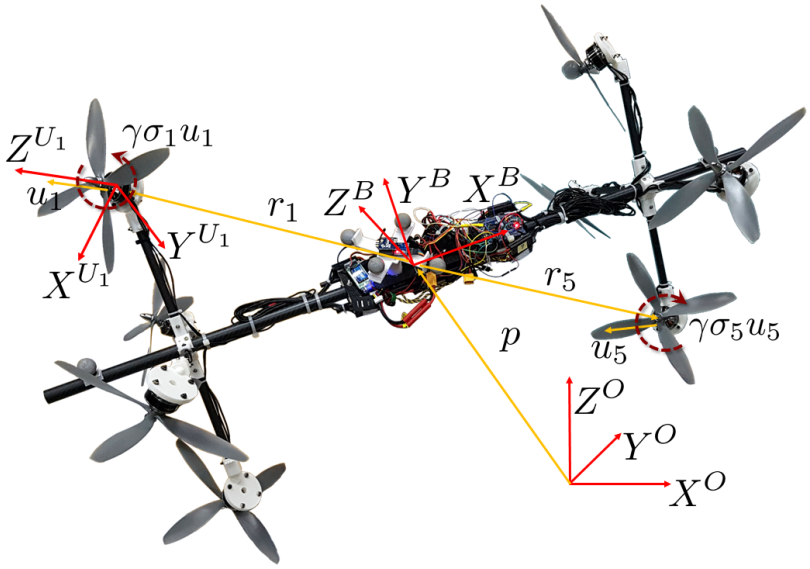

Figure 2-1: ODAR (omni-directional aerial robot) system with eight non-aligned bi-directional rotors as obtained from the design optimization (2.4). Also shown are the inertial, body and i-th rotor coordinate frames, $\{O\} := \{X^O, Y^O, Z^O\}$, $\{B\} := \{X^B, Y^B, Z^B\}$ and $\{U_i\} := \{X^{U_i}, Y^{U_i}, Z^{U_i}\}$, $i \in \{1, ..., 8\}$; and the orientation, position and reaction momentum vectors of i-th rotor, $u_i \in \mathrm{S}^2$, $r_i \in \Re^3$ and $\gamma\sigma_i u_i \in \Re^3$.

yet, suffers from the following two crucial limitations stemming from the under-actuation of the drone (i.e., cannot control its position and orientation at the same time with all axes of rotors parallel): 1) it would not be able to maintain contact or continue manipulation task when there blows side-way gust, since it cannot hold its orientation in the presence of lateral disturbance, particularly given that the attached robotic arm is typically of only low-DOF due to the payload limitation of the drone; and 2) it cannot exert downward force larger than its own weight and can do so only by turning off its rotors, since, for typical drones, all the rotors are aligned upward and driven by uni-directional ESCs, although this downward force is very important for many practical applications. These limitations, we believe, are because the multi-rotor drone platforms are designed/optimized for ease of flying, not for manipulation.

To overcome these limitations of the conventional multi-rotor drones, in this thesis, we propose a novel flying platform for aerial manipulation, **ODAR (omni-directional aerial robot)**, as shown in Fig. 2-1. By utilizing opportunistically aligned/distributed bi-directional rotors (with reversible ESC and bi-directional props), this system can attain omni-directional motion (i.e., arbitrary position/orientation) or produce omni-directional wrench (i.e., arbitrary force/torque). This full-actuation in SE(3) allows for such practically useful behaviors not possible with other typical aerial operation systems: 1) exerting force/torque in all directions, particularly pushing from the top (e.g., structure maintenance/repair); 2) pointing to any direction, while maintaining its posture against side-way wind (e.g., 360° camera shooting, fire-fighting hose operation); and 3) flying while adjusting its attitude at the same time (e.g., navigation in a pipe-cluttered environment). The key challenge for this system is the very tight thrust margin and weight budget under the currently-available motor and battery technologies, while being required to be fully-functioning with all the components on-board and no power cable tethered to the ground. For this, on top of using light-weight/stiff carbon-fiber structures and weight-reducing 3D-printed parts, we propose a general design optimization frame-

work for the ODAR system, which optimizes the pose of the rotors to maximize the minimum-guaranteed omni-directional force and torque generation with such an important aspect as inter-rotor aero-dynamic interference and task-specific anisotropy incorporated. We also propose a novel selective mapping algorithm to substantially subdue the destabilizing effect of "ESC-induced singularity", i.e., temporary loss of thrust when the reversible rotor changes its rotating direction due to the lack of position sensing (i.e., sensor-less BLDC), which, if not treated properly, can render the ODAR system behavior fairly shaky, unstable, and even resulting in crash. With all these implemented, we also perform experiments, in which our eight-rotor ODAR system can exert downward force larger than 60N much larger than its own weight (around 2.6kg); can control its tip position and force at the same time (i.e., hybrid position/force control) while fixing its attitude; and can even attain peg-in-hole task of its circular bar-end of 20mm diameter into 21mm diameter hole via bilateral teleoperation. All these levels of aerial manipulation performance, we believe, are reported by us in [26] for the first time.

There have been proposed several new designs of flying platforms to overcome the issue of under-actuation (e.g., [10, 11, 12, 13, 27, 16, 17, 18, 19, 20]). The works of [10, 11, 12, 13] advocate the use of extra actuators to tilt the direction of some or all of the rotors to overcome the issue of under-actuation. However, adding those extra actuators, possibly as many as the rotors, can substantially increase the system complexity and also result in further reduction of the already fairly-tight payload of the systems. The work of [27] presents a new aerial platform, so called, SmQ (spherically-connected multi-quadrotor) system, which is actuated by multiple drones connected by passive spherical joints to the platform, thereby, can deal with both the under-actuation and payload problems. This SmQ system, yet, still cannot exert downward force larger than its own weight due to the uni-lateral thrust generation of the standard drones. More closely related to our proposed ODAR system are the designs of [16, 17, 18, 19, 20], where rotors attached with non-parallel directions are used to attain the full-actuation on SE(3) with no extra

actuation. The works of [16, 17, 18] however optimize (or adjust) only the direction of the six rotors in S^1, while leaving their positions to be the same as those of the standard hexarotors. Although their designs achieve the full-actuation on SE(3), since their search space (i.e., S^1 of each rotor) is much narrower than ours (i.e., $\Re^3 \times S^2$ of each rotor), given the tight weight budget and thrust margin of currently-available motor and battery technologies, they would generate much less force/thrust omni-directionally than our ODAR system, which may be adequate for just standalone flying in mild environment (e.g., micro-gravity [18]), yet, likely substantially lacking for the manipulation tasks as demonstrated in this thesis. The work of [19] optimizes both the S^2-orientation and the position of the rotors as done here, yet, their goal is not to maximize the wrench generation, but to minimize the system size under the full-actuation constraint. Thus, similar as for [16, 17, 18], its wrench generation would be likely deficient for the manipulation tasks of this thesis. In fact, the implementation of this design [19] (and also [17, 18]) has not been reported yet. Most closely-related to our ODAR system is the design of [20], which also maximizes the omni-directional wrench generation. However, they do not take into account the inter-rotor aerodynamic interference, which not only significantly affects the rotor performance, but also results in infeasibility of their design, i.e., positions of some rotors are overlapped, thus, heuristically relocated to some vertices of a cube in [20]. In contrast, we optimize both the position and the orientation of each rotor, while fully incorporating such important aspects as the inter-rotor aero-interference and gravity compensation. Furthermore, to our knowledge, the level of performance of the aerial manipulation tasks demonstrated in this thesis (e.g., maximum downward force larger than 60N; aerial peg-in-hole with radial tolerance of 0.5mm) is unprecedented. We also believe that the issue of ESC-induced singularity is addressed by us in [26] for the first time with the selective mapping algorithm to substantially alleviate its destabilizing effect.

2.2 Mechanical Design

2.2.1 Design Description

We design the ODAR system to be of the bar shape (see Fig. 2-1), since: 1) it can conveniently hold or attach on itself tools commonly-used for many operation and manipulation tasks (e.g., screw-driver, drill, inspection probe, etc.) while effectively resisting the reaction moment of the tool through its longitudinal length; and 2) it can also mitigate the ground effect stemming from the fluid-structure interaction when the task takes place in a proximity of structures (e.g., close to wall, under the bridge girder, etc.), as the half of the actuators are located far from the interacting plane, thus, can still produce ample correcting wrench to subdue such ground effect (see Sec. 2.5.2). Of course, depending on task objectives, other shapes (e.g., spherical or disc shapes) would be desirable, for which the framework proposed in this chapter can also be applied.

To construct the ODAR system, with an inspiration from the design of conventional drones, we adopt each pair of two symmetrically-attached rotors as the basic actuator unit. More precisely, see Fig. 2-1, where the rotors 1 and 5 constitute such an actuator unit, with their rotor directions $u_1 = u_5 \in \mathrm{S}^2$ to be the same and their attachment locations $r_1 = -r_5 \in \Re^3$ symmetric w.r.t. the mainframe origin, while they rotate in different direction (e.g., their rotor types $\sigma_1 = 1$ and $\sigma_5 = -1$). With this symmetry, each rotor pair can then generate one-dimensional control force (e.g., $(\lambda_1 + \lambda_5)u_1$, where λ_i is the rotor thrust output) and one-dimensional control torque (e.g., $(\lambda_1 - \lambda_5)[r_1 \times u_1 + \gamma\sigma_1 u_1]$, where $\gamma > 0$ is the thrust-yaw ratio caused by the drag force), independently and separately. This then implies that we can render the ODAR system fully-actuated on SE(3) by using the three rotor pairs. This adoption of the rotor pairs as actuator units turns out to significantly simplify the process of design optimization (Sec. 2.2.2).

For the ODAR system to be omni-directional, we also adopt reversible ESCs

(electronic speed controllers) with the reversible propellers composed of two uni-directional props (i.e., with four blades) stacked together in the opposite direction (see Fig. 2-1). We also experimentally checked (see Fig. 2-9) that our stacked props, even with the inter-props flow interference, can still retain about 92% of the thrust production capability of a single uni-directional prop (with two blades). This reversible thrust generation is crucial particularly for aerial manipulation, since, only with that, we can exert downward pushing force larger than the weight of the system itself, an impossible feat with typical multi-rotor drones with uni-directional rotors.

One of the foremost challenges of the ODAR design is that, under the current available motor and battery technologies, the weight-thrust budget of the ODAR is fairly tight, particularly for untethered operation. This in fact spurs us to adopt the eight-rotor design of Fig. 2-1 for untethered operation instead of the six-rotor design for tethered operation in [28], since, with batteries, electronics, cables, etc., all on-board, we could not find some commercially available rotor-battery combination to fly our ODAR system with enough omni-directional wrench-exerting capability. This eight-rotor design provides the actuation redundancy, which can be utilized, e.g., to better allocate control actuation to each rotor or to ameliorate the issue of zero-crossing of the reversible ESC (see Sec. 2.4). With this tight weight-thrust budget constraint, it turns out to be of paramount importance to optimize the pose of the rotors as best as possible to maximize omni-directional wrench generation, which is the topic of the next Sec. 2.2.2.

2.2.2 Wrench-Maximizing Design Optimization

The goal of our design optimization here is to decide the attaching location $r_i \in \Re^3$ and the thrust generation direction $u_i \in \mathrm{S}^2$ of each rotor, $i = 1, .., n$, all expressed in the body frame $\{B\}$ (see Fig. 2-1 for a frame definition), to maximize omni-directional wrench generation. See Fig. 2-1. Using each pair of symmetrically-

attached rotors as an actuator unit as stated in Sec. 2.2.1, we first define the sets of u_i and r_i s.t.,

$$\mathcal{U} := \{u_i \in \mathrm{S}^2 \mid u_j = u_{j+\frac{n}{2}},\ i = 1, ..., n, j = 1, ..., \tfrac{n}{2}\}$$
$$\mathcal{R} := \{r_i \in \mathcal{R}_{\max} \mid r_j = -r_{j+\frac{n}{2}},\ i = 1, ..., n, j = 1, ..., \tfrac{n}{2}\}$$

where n is the total number of rotors, which is assumed to be even; and $\mathcal{R}_{\max}$ is the maximum allowable volume for all the rotor locations defined by

$$\mathcal{R}_{max} := \{r \in \Re^3 \mid \sqrt{r_y^2 + r_z^2} \leq R_{\max},\ |r_x| \leq \tfrac{L_{\max}}{2}\}$$

where $r = [r_x; r_y; r_z]$ expressed in $\{B\}$, and $R_{\max}$ and $L_{\max}$ are the maximum radius and length of the bar shape ODAR system. The type of rotors (i.e., left-handed or right-handed) is also considered as the optimization variable, since it affects the control torque generation via drag-induced reaction moment. For this, we define the set of rotor types as the optimization variable s.t.,

$$\mathcal{S} := \{\sigma_i \in \{1, -1\} |\ \sigma_j = -\sigma_{j+\frac{n}{2}},\ i = 1, ..., n, j = 1, ..., \tfrac{n}{2}\}$$

where $\sigma_i = 1$ means that the rotor generates upward thrust when rotating in clockwise direction (i.e., left-handed); and $\sigma_i = -1$ when in counter-clockwise direction (i.e., right-handed). The search space of our design optimization is then given by $\mathcal{U} \times \mathcal{R} \times \mathcal{S}$.

Let us denote the torque generation of the j-th rotor with the unit thrust generation (i.e., $\lambda_j = 1$) by

$$t_j := r_j \times u_j + \gamma \sigma_j u_j \in \Re^3 \tag{2.1}$$

where $\gamma \approx 0.02$ for our case, according to experiments. We can then see that, under the definitions of $(\mathcal{U}, \mathcal{R}, \mathcal{S})$ above, $t_j = -t_{j+n/2}$, implying that each pair of the rotors

j and $j+\frac{n}{2}$ can produce the one-dimensional control force $(\lambda_j + \lambda_{j+\frac{n}{2}}) \cdot u_j$ and the one-dimensional control torque $(\lambda_j - \lambda_{j+\frac{n}{2}}) \cdot t_j$ independently by adjusting their thrust outputs λ_j and $\lambda_{j+\frac{n}{2}}$, as stated in Sec. 2.2.1. With this design symmetry, the search space for the optimization becomes square root of the non-symmetric one, thereby, substantially reducing the complexity of solving the design optimization.

Our design optimization then boils down to the problem of finding (u_j, r_j, σ_j) of each rotor pair to maximize the feasible control force volume $\mathcal{V}_\mathcal{F}$ and feasible control torque volume $\mathcal{V}_\mathcal{M}$ defined as follows:

$$\mathcal{V}_\mathcal{F} := \{f \in \Re^3 \mid f = \sum_{i=1}^{n} \lambda_i u_i \ , \ \lambda_{\min} \leq \lambda_i \leq \lambda_{\max}\}$$
$$\mathcal{V}_\mathcal{M} := \{\tau \in \Re^3 \mid \tau = \sum_{i=1}^{n} \lambda_i t_i \ , \ \lambda_{\min} \leq \lambda_i \leq \lambda_{\max}\}$$

where λ_i is the thrust output of the i-th rotor; $\lambda_{\min}, \lambda_{\max} \in \Re$ are the minimum and maximum thrust for each rotor with $\lambda_{\max} = -\lambda_{\min} \geq 0$. Here, we assume uniformity among all the rotors. The sets $\mathcal{V}_\mathcal{F}$ and $\mathcal{V}_\mathcal{M}$ are both convex, since $\lambda_i u_i$ and $\lambda_i t_i$ each constitutes a convex set. See Fig. 2-2.

The ODAR system is purposed to be omni-directional. Thus, it is desired to maximize the minimum force and torque generation by the system for any attitude. Of particular importance is to generate force for any attitude larger than its own weight so that the system can fly in any attitude. For this, we define the guaranteed minimum control force for any orientation (generated collectively by all the rotors) s.t., with $\mathcal{N}_h := \{1, 2, ..., n/2\}$,

$$\mathcal{F}_{\min}(\mathcal{U}) := \min_{i,j \in \mathcal{N}_h} \sum_{k \in \mathcal{N}_h} 2\lambda_{\max} \frac{|(u_i \times u_j)^T u_k|}{||u_i \times u_j||} \tag{2.2}$$

which is the maximum radius of spheres centered at the origin and fully contained within the volume $\mathcal{V}_\mathcal{F}$. More specifically, consider the plane on $\mathcal{V}_\mathcal{F}$ spanned by u_i, u_j. Then, similar to the procedure developed for cable-driven robots in [29], the distance

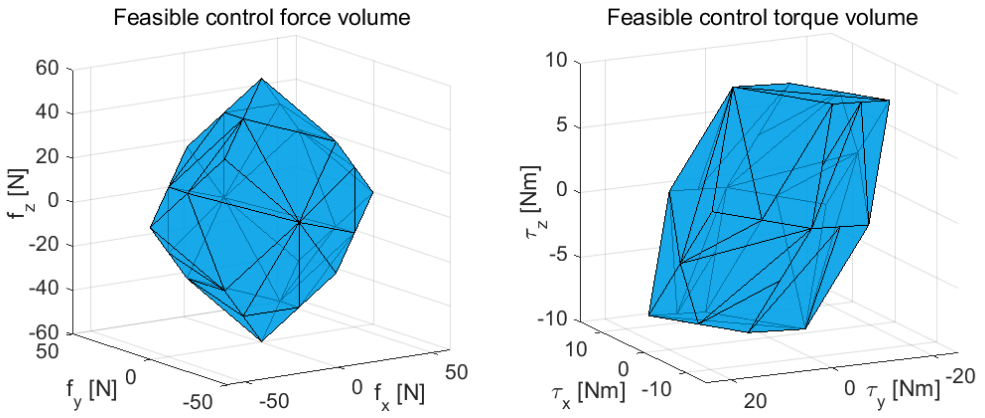

Figure 2-2: Feasible control force and torque volume ($\mathcal{V}_{\mathcal{F}}$,$\mathcal{V}_{\mathcal{M}}$) of the optimally designed eight-rotor ODAR system in Fig. 2-1.

from the origin to this plane along its normal vector $u_i \times u_j$ can be written as: with $i, j \in \mathcal{N}_h$,

$$d^{\mathcal{F}}_{F_{ij}} = \sum_{k \in S_{ij}} 2\lambda_{\max} \frac{(u_i \times u_j)^T u_k}{||u_i \times u_j||} + \sum_{k \in \tilde{S}_{ij}} 2\lambda_{\min} \frac{(u_i \times u_j)^T u_k}{||u_i \times u_j||}$$

where $S_{ij}, \tilde{S}_{ij}$ are defined by

$$S_{ij} := \{k | (u_i \times u_j)^T u_k \geq 0 \ , \ k \in \mathcal{N}_h\}, \quad \tilde{S}_{ij} := \mathcal{N}_h \setminus S_{ij}$$

and the multiplication by 2 of the RHS (right hand side) in the expression of $d^{\mathcal{F}}_{F_{ij}}$ is from our adoption of the rotor pairs. Then, since $\mathcal{V}_{\mathcal{F}}$ is convex with the origin in its interior as stated above, and further, symmetric w.r.t. the origin due to $\lambda_{\max} = -\lambda_{\min}$, we have $\mathcal{F}_{\min}(\mathcal{U}) = \min_{i,j \in \mathcal{N}_h} d^{\mathcal{F}}_{F_{ij}}$, from which (2.2) follows.

Nominally, the ODAR system is aimed to be omni-directional. However, depending on task objectives or its shape, it may be more advantageous to endow it with the ability of anisotropic force/torque generation ability so that its performance along the more often-used attitude is enhanced while that for the less-trotted attitude relaxed. For instance, the bar shape of our ODAR system naturally leads to the idea of using it more often with its pitch-yaw rotations (i.e., orientation about Y-axis or

Z-axis in Fig. 2-1) instead of with its roll-rotation. In this case, it would be more desirable to "shape" the force generation capability in such a way that the force generation is maximized in the XZ-plane (i.e., sagittal plane), while relaxing along the Y-axis expressed in $\{B\}$. Note that, even so, such roll-directional operations as screw-driver or drilling can still (and more conveniently) be achieved by simply attaching a rotating-tool with reaction moment succumbed by the ODAR control torque generation.

This "anisotropic shaping" of the force generation can be attained by using the following weighted $\mathcal{F}^W_{\min}(\mathcal{U})$ in the place of $\mathcal{F}_{\min}(\mathcal{U})$ in (2.2):

$$\mathcal{F}^W_{\min}(\mathcal{U}) := \min_{i,j\in\mathcal{N}_h} \sum_{k\in\mathcal{N}_h} 2\lambda_{\max} \frac{|(W^{-1}u_i \times W^{-1}u_j)^T W^{-1}u_k|}{||W^{-1}u_i \times W^{-1}u_j||}$$

where $W := \mathrm{diag}[W_x, W_y, W_z]$ is the weight matrix with $0 < W_\star \leq 1$ (with (x, y, z) corresponding to (X, Y, Z) of $\{B\}$). Here, note that, if $W_\star < 1$, $\mathcal{V}_\mathcal{F}$ will be stretched by $1/W_\star$ along that direction, thus, $\mathcal{F}^W_{\min}(\mathcal{U})$ will be strengthened along that direction as compared to $\mathcal{F}_{\min}(\mathcal{U})$, thereby, relaxing the force generation requirement along that direction. For instance, for our bar shape ODAR, we choose $W = [1, 0.4, 1]$ (see Table 1) so that the force generation requirement along the body-fixed E-axis is relaxed, while retaining that for the XZ-plane. Although the ODAR system can still operate with $W = [1, 1, 1]$, we however found this $W = [1, 0.4, 1]$ provides us a better-tuned ODAR system for the operations with more pitch/yaw-rotations as experimented in Sec. 2.5.2.

On the other hand, for the control torque generation optimization, similar to (2.2), we can define the guaranteed minimum control torque for all the orientation (generated collectively by all the rotors) s.t.,

$$\mathcal{M}_{\min}(\mathcal{U}, \mathcal{R}, \mathcal{S}) := \min_{i,j\in\mathcal{N}_h} \sum_{k\in\mathcal{N}_h} 2\lambda_{\max} \frac{|(t_i \times t_j)^T t_k|}{||t_i \times t_j||} \tag{2.3}$$

or its weighted version $\mathcal{M}^V_{\min}(\mathcal{U}, \mathcal{R}, \mathcal{S})$ similar to $\mathcal{F}^W_{\min}$ above with the weight matrix

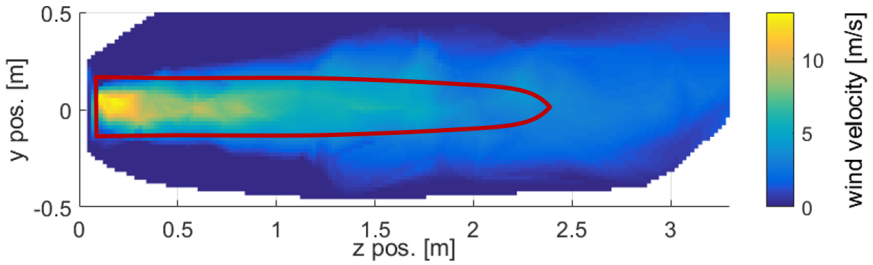

Figure 2-3: Anemometer measurement of wind velocity distribution downstream the rotor generating thrust required for hovering with the rest pose ($R_{OB} = I$) with the r_a-function also marked with interference-threshold wind speed to be 4m/s.

$V := \text{diag}[V_x, V_y, V_z]$ to attain the anisotropic torque generation capability. For our design below, we choose $V = [1, 1, 1]$.

We can then formulate the design optimization problem as a constrained optimization problem for (u_i, r_i, σ_i) s.t.,

$$\max_{\mathcal{R},\mathcal{S}} \ \mathcal{M}^V_{\min}(\arg\max_{\mathcal{U}} \mathcal{F}^W_{\min}(\mathcal{U}), \mathcal{R}, \mathcal{S}) \tag{2.4}$$

$$\text{subj. to} \ \ u_i^T u_i = 1, \quad r_i \in \mathcal{R}_{max} \tag{2.5}$$

$$\mathcal{F}^W_{\min}(\mathcal{U}) \geq mg, \quad d_{\text{aero}}(\mathcal{U}, \mathcal{R}) \geq D \tag{2.6}$$

where (2.5) are to constrain $u_i \in \text{S}^2$ and to constrain the volume of the ODAR system; (2.6) are to ensure that the ODAR system can fly while overcoming its own weight (with the relaxation endowed by $W = [W_x, W_y, W_z]$ as discussed above), while reducing the inter-rotor aerodynamic interference by ensuring that the gap between the flow stream of each rotor (i.e., $\mathcal{C}_{a,i}$) to other rotors (i.e., r_j) be larger than a certain value D, i.e., $d_{\text{aero}}(\mathcal{U}, \mathcal{R}) := \min_{i,j} ||c_{a,i} - r_j|| \geq D, \quad c_{a,i} \in \mathcal{C}_{a,i}$, where $i, j \in \{1, 2, ..., n\}$, $i \neq j$ and $\mathcal{C}_{a,i}(r_i, R_{BU_i}) := \{c \in \Re^3 | c = R_{BU_i} v + r_i, \ \sqrt{v_x^2 + v_y^2} \leq r_a(v_z)\}$ is the flow stream volume of the i-th rotor, with $R_{BU_i} \in \text{SO}(3)$ being the rotation matrix from $\{B\}$ to $\{U_i\}$ (see Fig. 2-1), $v := [v_x; v_y; v_z] \in \Re^3$ a position vector in $\{U_i\}$, and $r_a : \Re \to \Re$ being the axial shape function of the aerodynamic space also similarly used in [19]. To identify this flow-axial function r_a, we perform an experiment to measure wind velocity distribution downstream the rotor generating

Table 2.1: Design optimization parameters & optimized values

Symbols	Values	Units
n	8	-
$R_{\max}$	0.24	m
$L_{\max}$	0.8	m
W	$\text{diag}[1, 0.4, 1]$	-
V	$\text{diag}[1, 1, 1]$	-
D	0.12	m
$\mathcal{F}_{\min}(\mathcal{U})$	$2.04\lambda_{\max}$	N
$\mathcal{F}^{xz}_{\min}(\mathcal{U})$	$3.85\lambda_{\max}$	N
$\mathcal{M}_{\min}(\mathcal{U}, \mathcal{R}, \mathcal{S})$	$0.894\lambda_{\max}$	Nm
$d_{\text{aero}}(\mathcal{U}, \mathcal{R})$	0.125	m
$\mathcal{U}$	$u_1 = [0.68; 0.28; 0.68]$ $u_2 = [0.68; 0.28; -0.68]$ $u_3 = [0.68; -0.28; 0.68]$ $u_4 = [0.68; -0.28; -0.68]$	-
$\mathcal{R}$	$r_1 = [0.40; -0.17; 0.17]$ $r_2 = [0.40; 0.17; 0.17]$ $r_3 = [0.40; -0.17; -0.17]$ $r_4 = [0.40; 0.17; -0.17]$	m
$\mathcal{T}$	$t_1 = [-0.15; -0.15; 0.24]$ $t_2 = [-0.18; 0.38; 0.01]$ $t_3 = [-0.18; -0.38; -0.01]$ $t_4 = [-0.18; 0.16; -0.21]$	m
$\mathcal{S}$	$\sigma_1 = +1, \quad \sigma_2 = -1,$ $\sigma_3 = -1, \quad \sigma_4 = -1$	-

thrust required for hovering with the rest pose ($R_{OB} = I$). See Fig. 2-3, from which we obtain r_a-function to be a tapered cylinder as marked therein. For this, we choose the interference-threshold wind velocity to be 4 m/s (i.e., gentle breeze according to Beaufort scale, known to be adequate for drone flying).

We solve this constrained optimization (2.4)-(2.6) and the obtained optimal ODAR design is illustrated in Fig. 2-1 with its feasible control force and torque volumes $\mathcal{V}_{\mathcal{F}}$ and $\mathcal{V}_{\mathcal{M}}$ also shown in Fig. 2-2. Since the optimization problem (2.4)-(2.6) is non-convex and has complex form of objective and constraints, the solution

is obtained with the grid search method. Note also from (2.4) that we first determine the thrust directions $u_i \in \mathrm{S}^2$ of all the rotors, and then solve for their attaching location $r_i \in \Re^3$ and their types σ_i given the obtained $u_i \in \mathrm{S}^2$. This *sequential formulation* turns out to significantly speed up the solving process of the optimization (2.4)-(2.6) while still providing an adequate design as experimentally validated in Sec. 2.5.

Design parameters and optimized design variables are summarized in Table 2.1, where only those of the rotors $1, 2, 3, 4$ are given due to the symmetric design of the ODAR system. For this optimization, we also assume $\lambda_{\max} = 9.7$N according to the specification of the rotors used in the implementation (see Sec. 2.5.1). With this rotor thrust, the ODAR system can overcome its weight along the XZ-plane, as the minimum guaranteed control force within this sagittal XZ-plane is $F^{xz}_{\min}(\mathcal{U}) = 37.35$N, whereas the weight of the ODAR system is 25.48N (i.e., 2.6kg - see Sec. 2.5.1). Here, note that the omni-directionally guaranteed minimum force $F_{\min}(\mathcal{U}) = 17.97$N is less than the weight of the system, even if we enforce (2.6). This is because we use the weight $W = [1, 0.4, 1]$ to relax the force generation along the body-fixed Y-axis as our ODAR system will be used mostly with minimal roll-rotation as stated above and also experimentally validated in Sec. 2.5.2. With $W = [1, 1, 1]$, we can ensure $F_{\min}(\mathcal{U}) > 25.48$N, yet, with deterioration of the force generating capability in the sagittal plane. See Fig. 2-2 also for the feasible control force and torque volumes $\mathcal{V}_{\mathcal{F}}$ and $\mathcal{V}_{\mathcal{M}}$ with the metric information endowed by $\lambda_{\max}$ with the omni-directionally guaranteed minimum control torque to be 8.67Nm.

2.3 System Modeling and Control Design

2.3.1 System Modeling

With the design optimization of Sec. 2.2.2, that determines u_i, r_i, σ_i to guarantee the minimum force $\mathcal{F}_{\min}$ and torque $\mathcal{M}_{\min}$ for any attitude, we can model the ODAR

system as a fully-actuated rigid body s.t.,

$$\begin{aligned} m\ddot{p} &= R_{OB}B_f\lambda - mge_3 + f_e \\ J\dot{\omega} + \omega \times J\omega &= B_\tau\lambda + \tau_e \end{aligned} \tag{2.7}$$

where $m > 0$ is the mass with $p \in \Re^3$ being the center-of-mass position expressed in $\{O\}$, $J \in \Re^{3\times3}$ is the inertia matrix with $\omega \in \Re^3$ being the angular velocity expressed in $\{B\}$, $R_{OB} \in \mathrm{SO}(3)$ is the rotation matrix from $\{O\}$ to $\{B\}$, and g is the gravitational constant with $e_3 = [0; 0; 1]$. See Fig. 2-1. Also, $\lambda = [\lambda_1; \lambda_2; \cdots; \lambda_n] \in \Re^n$ is the collection of the thrust inputs of all rotors, and $B_f, B_\tau \in \Re^{3\times n}$ are the mapping matrices from the thrust inputs to the control force and control torque, with their i-th column respectively specified by

$$B_f^i := u_i, \qquad B_\tau^i := t_i, \quad i = 1, ..., n$$

where $u_i, t_i \in \Re^3$ are the one-dimensional space of the thrust and torque generation of the i-th rotor - see Sec. 2.2.2.

For notational convenience, we define

$$\begin{pmatrix} f \\ \tau \end{pmatrix} := \begin{bmatrix} R_{OB} & 0 \\ 0 & I_{3\times3} \end{bmatrix} \begin{bmatrix} B_f \\ B_\tau \end{bmatrix} \lambda =: \tilde{R}B\lambda \tag{2.8}$$

where $f, \tau \in \Re^3$ are the translation and orientation controls respectively expressed in $\{O\}$ and $\{B\}$, $\tilde{R} := \mathrm{diag}[R_{OB}, I_{3\times3}] \in \Re^{6\times6}$, and $B := [B_f; B_\tau] \in \Re^{6\times n}$ is the mapping matrix, which is to be full row rank, as the design optimization in Sec. 2.2.2 enforces the full-actuation of the ODAR system. Thus, from now on, we directly design (f, τ) for various control objectives assuming that they are arbitrarily assignable.

2.3.2 Pose Trajectory Tracking Control

For the pose tracking control of the ODAR system in SE(3), we design f, τ to be proportional-integral-derivative (PID) controls in E(3) and SO(3) as follows:

$$f = m\ddot{p}_d - k_d\dot{e}_p - k_p e_p - k_I \int_0^t e_p ds + mge_3$$
$$\tau = J\dot{\omega}'_d + \omega'_d \times J\omega'_d - k_\omega(\omega - \omega'_d) - k_R e_R - k_{R,I} e_{R,I}$$

where $p_d(t) \in \Re^3$ and $R_d(t) \in \mathrm{SO}(3)$ are the desired position and attitude trajectory expressed in $\{O\}$; $e_p := p - p_d \in \Re^3$; $\omega_d := (R_d^T \dot{R}_d)^\vee$ is the desired angular velocity expressed in $R_d(t)$-frame, where $(\bullet)^\vee$ is the vee operator, that maps so(3) to $\Re^3$ [30]; $\omega'_d := R_{OB}^T R_d \omega_d \in \Re^3$ and $\dot{\omega}'_d := R_{OB}^T R_d \dot{\omega}_d \in \Re^3$ are the desired angular velocity and acceleration expressed in $\{B\}$; $e_R := \frac{1}{2} k_R (R_d^T R_{OB} - R_{OB}^T R_d)^\vee \in \Re^3$ is the attitude error vector; $e_{R,I} := \int_0^t (\omega - \omega'_d + c_2 e_R) ds \in \Re^3$ is the error integral in SO(3) following [31] with c_2 also decided as in [31]; and $k_d, k_p, k_I \in \Re^{3\times3}$, $k_\omega, k_R, k_{R,I} \in \Re$ are the diagonal positive-definite and positive control gains. Here, we adopt the attitude PID-control in SO(3) of [31] instead of that based on the Euler angle representation, since the ODAR system is omni-directional and can assume large rotation, thus, it is important to avoid singularity in SO(3). This pose tracking control can also be serve as the basis for impedance control, as the ODAR system is back-drivable.

2.3.3 Hybrid Pose/Wrench Control

For a manipulation to the external environment using the ODAR system, hybrid position/force controller is designed which enables the contact force in constrained space and position in unconstrained space independently, while exploiting momentum-based observer [32, 33] to estimate the external force applied to the system. Also, as the system has back-drivable property with thrust-propelled actuators, a compliance given by this property can be exploited for the interaction/compliant control as well.

In case of a certain kinematic constraint applied to the system, the dynamics (2.7) can be rewritten with a combined form as

$$\begin{aligned} M\ddot{q} + C(q,\dot{q})\dot{q} + G(q) + A^T(q)\lambda_c &= w + w_e \\ A(q)\dot{q} &= 0 \end{aligned} \tag{2.9}$$

where $\dot{q} := [\dot{p}; \omega] \in \Re^6$, $M := \mathrm{diag}[mI_{3\times3}, J]$, $C := \mathrm{diag}[0_{3\times3}, -(J\omega)^\wedge]$, $(\bullet)^\wedge$ is a map from $\Re^3$ to so(3), $G(q) := [mge_3; 0_{3\times1}]$, $A(q) \in \Re^{n_p\times6}$ is a Pfaffian constraint matrix, n_p is the number of constraint, $w := \tilde{R}B\lambda = [f; \tau]$ is the control wrench, $w_e := [f_e; \tau_e] \in \Re^6$ is the external wrench and $\lambda_c \in \Re^{n_p}$ is a lagrange multiplier that physically means contact force for each constrained direction. Then with the new variable $\phi \in \Re^{6-n_p}$ defined as $\dot{q} = \Delta(\phi)\dot{\phi}$ where $\Delta(\phi) \in \Re^{6\times(6-n_p)}$ is a matrix that satisfies $A(q)\Delta(\phi)$, reduced dynamics can be written as below

$$D_\phi(\phi)\ddot{\phi} + Q_\phi(\phi,\dot{\phi})\dot{\phi} + G_\phi(\phi) = \Delta^T w + \Delta^T w_e$$

where $D_\phi(\phi) := \Delta^T M\Delta \in \Re^{(6-n_p)\times(6-n_p)}$, $Q_\phi(\phi,\dot{\phi}) := \Delta^T(M\dot{\Delta}+C\Delta) \in \Re^{(6-n_p)\times(6-n_p)}$ are inertia and coriolis matrix of the reduced dynamics, $G_\phi(\phi) := \Delta^T G(q) \in \Re^{6-n_p}$.

With this dynamics, a hybrid position/force controller that achieves the control objective of $e_\phi(t) \to 0$, $\lambda_c \to \lambda_{c,d}(t)$ can be designed as following

$$\begin{aligned} w = M\Delta(\phi)(\ddot{\phi}_d - K_d\dot{e}_\phi - K_p e_\phi - K_i e_{\phi,i}) \\ +[M\dot{\Delta}(\phi) + C(q,\dot{q})\Delta(\phi)]\dot{\phi} + G(q) \\ +A^T(q)[\lambda_{c,d} - K_f\int_0^t(\hat{\lambda}_c - \lambda_{c,d})ds] \end{aligned}$$

where $\ddot{\phi}_d := \Omega([\ddot{p}_d; \dot{\omega}'_d] - \dot{\Delta}\dot{\phi})$, $\dot{e}_\phi := \Omega[\dot{e}_x; \omega - \omega'_d]$, $e_\phi := \Omega[e_x; e_R]$, $e_{\phi,i} := \Omega[\int_0^t e_x ds; e_{R,i}] \in \Re^{6-n_p}$, $\Omega := (\Delta^T M\Delta)^{-1}\Delta^T M \in \Re^{(6-p)\times6}$ is the matrix that annihilates the orthogonal complement of $\Delta(\phi)$ w.r.t. M-metric, and $K_d, K_p, K_i \in \Re$, $K_f \in \Re^{p\times p}$ are the control gains. Here, for the estimate of the lagrange multiplier λ_c, momentum-

based observer [32, 33] is used with unconstrained dynamics considering the contact wrench $A^T(q)\lambda_c$ also as an external wrench. Then, the estimate $\hat{\lambda}_c$ can be obtained by projecting the estimated external wrench to the constrained space as the equation below.

$$\hat{w}_e = k_e \left(M\dot{q} - \int_0^t (w - C\dot{q} - G + \hat{w}_e) ds \right)$$
$$\hat{\lambda}_c = -(AA^T)^{-1} A\hat{w}_e$$

where $\hat{w}_e \in \Re^6$ is the estimated external wrench, $k_e \in \Re^{6\times 6}$ is the observer gain.

2.3.4 PSPM-Based Teleoperation

Although the motion and interaction controller in SE(3) is developed in previous subsections, it is still challenging to conduct complex or precision-requiring tasks (e.g., peg-in-hole) in a fully autonomous way. Therefore in this work, bilateral teleoperation is considered for the ODAR system so that the human sensory system can be utilized to enable tasks that are difficult with autonomous control. During the teleoperation, the command from the operator is not guaranteed to be always smooth, and some undesired command such as an abrupt change can even make the system unstable. To prevent this issue, the command from operator is modulated through the passive set-position modulation (PSPM) algorithm in [34, 35] with the extension to be used in SE(3). By the PSPM, the set pose given by the operator is modulated to the extent of satisfying passivity of the slave side while minimizing the difference with the original set pose.

First of all, when the desired position $x_d(k)$ and attitude $R_d(k)$ are given by the master device, the actuation of the slave or the ODAR system is defined as following.

$$f = -K_v\dot{p} - K_p(p - \bar{p}_d(k))$$
$$\tau = -K_\omega\omega - K_R \frac{\theta_R^t}{2\sin\theta_R^t} \left(\bar{R}_d^T(k) R_{OB} - R_{OB}^T \bar{R}_d(k) \right)^\vee$$

where $K_v, K_p, K_\omega, K_R \in \Re^{3\times 3}$ are the positive-definite diagonal gain matrices, and $\bar{p}_d(k) \in \Re^3$ is the modulated desire position from $x_d(k)$ obtained by PSPM. Also, $\bar{R}_d(k) \in \mathrm{SO}(3)$ is the modulated desired attitude from $R_d(k)$ and $\theta_R^t :=$ $\cos^{-1}\frac{1}{2}\left(\mathrm{tr}(\bar{R}_d^T(k)R_{OB})-1\right)$. Here, to obtain $\bar{R}_d(k)$, PSPM algorithm is extended to be also used in SO(3).

At each time when the desired position $x_d(k)$ is given, modulated position $\bar{p}_d(k)$ is defined by

$$\begin{aligned} &\min_{\bar{y}(k)} \ ||p_d(k) - \bar{p}_d(k)|| \\ &\text{subj. to} \ \ E_t(k) = E_t(k-1) + D_{t,\min}(k-1) - \Delta\bar{P}_t(k) \geq 0 \end{aligned}$$

where $E_t(k) \geq 0$ is the virtual energy reservoir for translation; $D_{t,\min}(k) :=$ $\frac{1}{T_{k+1}-T_k}\sum_{i=1}^{3} K_{v,i}\left(p_i^{\max}(k) - p_i^{\min}(k)\right)^2$ is the conservative approximation of dissipated energy by the damping K_v, $K_{v,i}$ is the i-th diagonal element of K_v, and $p_i^{\max}(k), p_i^{\min}(k)$ are the maximum and minimum of $p_i(t)$ during $[T_k, T_{k+1})$; and with $||x||_A^2 := x^T A x$, $\Delta\bar{P}_t(k) := \frac{1}{2}||p(T_k) - \bar{p}_d(k)||_{K_p}^2 - \frac{1}{2}||p(T_k) - \bar{p}_d(k-1)||_{K_p}^2$ is the energy jump caused by the change of set position. The optimization formulation means that $\bar{p}_d(k)$ is chosen to be as close as possible to $p_d(k)$ while satisfying the passivity constraint.

For the modulation of the desired attitude, similar procedure is conducted by extending the PSPM algorithm to SO(3) using the notion of SO(3) spring. When the desired attitude $R_d(k)$ is given, the modulated set rotation $\bar{R}_d(k)$ is defined by

$$\begin{aligned} &\min_{\bar{R}_d(k)} \ || \left(\log\{\bar{R}_d^T(k)R_d(k)\}\right)^\vee || \\ &\text{subj. to} \ \ E_r(k) = E_r(k-1) + D_r(k-1) - \Delta\bar{P}_r(k) \geq 0 \end{aligned}$$

where $\log\{R\} = \frac{\theta}{2\sin\theta}(R - R^T)$ and $\theta = \cos^{-1}\left(\frac{\mathrm{tr}(R)-1}{2}\right)$; $E_r(k) \geq 0$ is the virtual energy reservoir for rotation; $D_r(k) := \sum_{i=1}^{3} K_{\omega,i}|\omega_i|^2(T_{k+1} - T_k)$ is the dissipated

energy by the damping K_ω, $K_{\omega,i}$ is the i-th diagonal element of K_ω; and $\Delta\bar{P}_r(k) := \frac{1}{2}||\left(\log\{\bar{R}_d^T(k)R(T_k)\}\right)^\vee||_{K_R}^2 - \frac{1}{2}||\left(\log\{\bar{R}_d^T(k-1)R(T_k)\}\right)^\vee||_{K_R}^2$ is the jump of SO(3) spring energy caused by the change of set rotation. Here, to prevent the excessive cumulation of the energy which can be a potential danger, we make $E_t(k), E_r(k)$ cannot be greater than certain limit value $E_t^{\max}, E_r^{\max}$ by discarding the energy over the limit.

In this work, the PSPM algorithm is applied only for the slave side of the system. This application is sufficient for the system since: 1) we assume here that the time delay or data loss between master and slave is almost negligible (e.g., master and slave that located adjacently), the stability issue from the delay is not critical; 2) the stability of the master side can still be enforced by suitable selection of the feedback gains. Yet, in case that the delay/loss is not negligible, the PSPM can be applied for both master and slave side, guaranteeing passivity of the closed-loop system [34].

Then from the master side, the desired position and attitude can be mapped such as,

$$\begin{aligned} p_d(k) &= \eta p_h(k) + p_0 \\ R_d(k) &= R_h(k) \end{aligned}$$

where $\eta \in \Re$ is the scale factor between the master device and the slave robot, $p_h \in \Re^3$, $R_h \in \mathrm{SO}(3)$ are the position and the orientation of the master device, and $p_0 \in \Re^3$ is the position offset. Also, the haptic feedback for the operator is designed to be exerted with spring-damper connection and additional force feedback as

$$\begin{aligned} f_h &= -B_{h,t}\dot{p}_h - K_{h,t}(p_d - \bar{p}_d) + K_{h,f}\hat{f}_e \\ \tau_h &= -B_{h,r}\omega_h - K_{h,r}\left(\bar{R}_d^T R_d - R_d^T \bar{R}_d\right)^\vee + K_{h,\tau}\hat{\tau}_e \end{aligned}$$

where $B_{h,t}, K_{h,t}, B_{h,r}, K_{h,r}, K_{h,f}, K_{h,\tau} \in \Re^{3\times 3}$ are diagonal gain matrices, $f_h, \tau_h \in \Re^3$ are the force/torque feedback realized at the haptic device, and $\hat{f}_e, \hat{\tau}_e \in \Re^3$ are the

estimated or measured external force and torque applied to the system.

2.4 Control Allocation with Selective Mapping

2.4.1 Infinity-Norm Minimization

Once the desired control wrench $w := [f; \tau] \in \Re^6$ is calculated as stated in Sec. 2.3, it needs to be distributed among the n-rotor commands $\lambda = (\lambda_1, \lambda_2, ..., \lambda_n) \in \Re^n$ to produce $U = B\lambda$, where $U := [R_{OB}^T f; \tau] = \bar{R}^{-1} w \in \Re^6$ from (2.8). Here, we assume $n \geq 8$ as stated in Sec. 2.2.1, thus, the actuation redundancy should be addressed as well. Most commonly-used method for this is to minimize the two norm of $\lambda \in \Re^n$ i.e., $\lambda_o = B^\dagger U$, where $B^\dagger := B^T(BB^T)^{-1} \in \Re^{n\times 6}$. Although also adopted in [20] and relevant to power efficiency to some extent, this two norm optimization is not so suitable for the ODAR system, since, with the rotor actuation margin fairly tight, only one rotor saturation can simply result in instability and even crash, which is not captured by the two norm optimization. To address this issue, here, we adopt the infinity-norm optimization for (2.8), which can be written by

$$\begin{aligned} \lambda_\alpha &:= B^\dagger U + N_B \xi' \\ \xi' &:= \underset{\xi \in \Re^{n-6}}{\arg\min} \; ||B^\dagger U + N_B \xi||_\infty \end{aligned} \tag{2.10}$$

where $\lambda_\alpha \in \Re^n$ is the thrust vector to be used, $N_B \in \Re^{n\times(n-6)}$ is the kernel of the mapping matrix B, and $\xi' \in \Re^{n-6}$ is the null vector component. This infinity-norm optimization has been extensively studied for many applications (e.g., for robot manipulators [36]), yet, is known in general not to assume a closed-form solution.

For the eight-rotor ODAR system optimally designed as shown in Fig. 2-1, it turns out that we can attain the closed-form of the infinity-norm optimization (2.10) with a slight modification of the design. More precisely, we change σ_1 in Table 1 from $\sigma_1 = 1$ to $\sigma_1 = -1$. With this $\sigma_1 = -1$, we then have the same op-

timal design of Table 1 for (2.4)-(2.6) , except $\mathcal{T} = \{t_1 = [-0.18; -0.16; 0.21], t_2 = [-0.18; 0.38; 0.01], t_3 = [-0.18; -0.38; -0.01], t_4 = [-0.18; 0.16; -0.21]\}$. With this modified design, the minimum guaranteed control force remains the same (since $\mathcal{U}$ is not changed), whereas the minimum guaranteed control torque is reduced only by 3.6%. Further, the design then satisfies the following condition with $\nu = [1; -1; -1; 1]$, which is used for the computation of a closed-form solution as summarized in the next Prop. 1:

$$\begin{aligned} &B_{f,\star}\nu = 0,\ B_{\tau,\star}\nu = 0,\ B_{f,r} = B_{f,l},\ B_{\tau,r} = -B_{\tau,l} \\ &\nu := [\nu_1; \nu_2; \nu_3; \nu_4]\ ,\ |\nu_i| = 1\ ,\ i \in \mathcal{N}_h \end{aligned} \tag{2.11}$$

where $\star \in \{r, l\}$ (with r-side with rotors 1-4 and l-side with rotors 5-8: see Fig. 2-1) and $B =: [B_{f,r}\ B_{f,l}; B_{\tau,r}\ B_{\tau,l}]$, $B_{f,\star}, B_{\tau,\star} \in \Re^{3\times 4}$.

Proposition 1. *Consider the mapping matrix $B \in \Re^{6\times 8}$ of* (2.8) *for the ODAR system with $n = 8$. If this B-matrix satisfies the properties* (2.11)*, the solution of the infinity-norm optimization* (2.10) *is given by*

$$\begin{aligned} \lambda_\alpha &= B^\dagger U + N_B \begin{bmatrix} \xi'_r \\ \xi'_l \end{bmatrix} \\ \xi'_\star &= -\tfrac{1}{2}(\lambda^\nu_{\star,\max} + \lambda^\nu_{\star,\min}) \end{aligned} \tag{2.12}$$

where $\star \in \{r, l\}$, $\lambda^\nu_\star := diag(\nu)\lambda_{o,\star} \in \Re^4$ with $\lambda_o := B^\dagger U = [\lambda_{o,r}; \lambda_{o,l}] \in \Re^8$; and $\lambda^\nu_{\star,\max}, \lambda^\nu_{\star,\min}$ are the maximum and minimum component of $\lambda^\nu_\star$.

Proof. Under the properties of (2.11), the kernel matrix of the B-matrix can be written as:

$$N_B = \begin{bmatrix} \nu & 0 \\ 0 & \nu \end{bmatrix} \in \Re^{8\times 2} \tag{2.13}$$

where $\nu = [1; -1; -1; 1] \in \Re^4$. We can then attain, from (2.12), that:

$$|\lambda_{\alpha,\star,i}| = |\lambda_{o,\star,i} + \nu_i \xi'_\star| = |\nu_i \lambda_{o,\star,i} + \xi'_\star| \tag{2.14}$$

where $\star \in \{r, l\}$, $i \in \mathcal{N}_h$, $\xi'_\star \in \Re$, and $\lambda_\alpha =: [\lambda_{\alpha,r}; \lambda_{\alpha,l}]$. Then, the infinity-norm of $\lambda_{\alpha,\star}$ is given by

$$\begin{aligned} ||\lambda_{\alpha,\star}||_\infty &= \max(|\max(\lambda_{\alpha,\star})|, |\min(\lambda_{\alpha,\star})|) \\ &= \max(|\lambda^\nu_{\star,\max} + \xi'_\star|, |\lambda^\nu_{\star,\min} + \xi'_\star|) \end{aligned}$$

which is minimized when $\xi'_\star$ is selected s.t.,

$$\lambda^\nu_{\star,\max} + \xi'_\star = -(\lambda^\nu_{\star,\min} + \xi'_\star)$$

Finally, note that $||\lambda_\alpha||_\infty = \max(||\lambda_{\alpha,r}||_\infty, ||\lambda_{\alpha,l}||_\infty)$, which is also minimized with the above selection of $\xi'_\star$, since the kernel matrix N_B has the block-diagonal structure and ξ'_r, ξ'_l only affects $||\lambda_{\alpha,r}||_\infty, ||\lambda_{\alpha,l}||_\infty$, respectively. □

We will use this solution of (2.12) as a basis of the control allocation for further adjustment of λ_i to mitigate the issue arising from the zero-crossing of the rotor as stated in the next Sec. 2.4.2.

2.4.2 ESC-Induced Singularity and Selective Mapping

Once we applied the allocated thrust input λ_α of (2.12) to the ODAR system for the hovering, at certain attitudes, the system behavior becomes fairly shaky and, in some cases, even goes unstable and results in crash. This we found stems from the phenomenon that the BLDC motors of the rotors slow-down and re-rotate when they are commanded to suddenly change their rotation direction - see Fig. 2-4, where the rotors "hesitate" when changing their rotating directions (around 9.5sec and 14.5sec). This hesitation is due to the lack of position sensing of the BLDC rotor

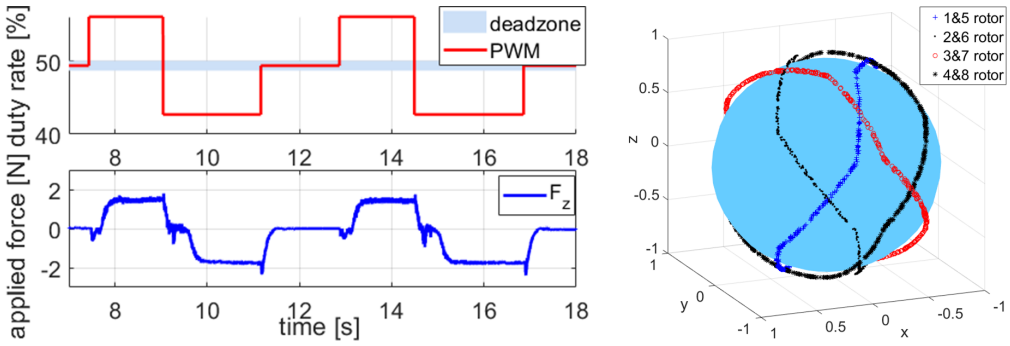

Figure 2-4: (Left) Plot of input PWM to the ESC and resultant force generated by bi-directional rotor when desired direction of force is suddenly changed (around 9sec and 14.5sec); (Right) Slicing of S^2-sphere by the zero-thrust lines of the rotor pairs of the eight-rotor ODAR system. Note that two pairs of the rotors simultaneously become zero-thrust only at certain points on S^2.

motors, that is, typical off-the-shelf drone rotor ESCs are not equipped with position sensors (e.g., hall sensor) and instead, rely on the back-EMF (electromotive force) to estimate the rotor position for their control, which becomes not so useful when the rotor speed gets close to zero. Typical ESCs in fact utilize their own certain "bootstrapping" algorithm to start rotation due to this lack of position sensing. Even though position sensors can be embedded into the rotor BLDC motors, in this thesis, we consider the case of the sensor-less BLDC rotors, since, to our knowledge, all the commercially-available drone ESCs are all sensor-less, likely due to the added complexity and cost of extra hall sensors.

Shown in Fig. 2-4 is the slicing of S^2 (pitch and yaw) by the zero-thrust lines of some rotor pairs when the eight-rotor ODAR of Fig. 2-1 stays hovering quasi-statically while changing its attitude, where the two rotors of each pair experience the zero thrust at the same time due to the symmetry of the ODAR design and the hovering operation. This Fig. 2-4 then shows that the zero-crossing of some rotors is likely inevitable for any omni-directional operation of the ODAR system, since the system needs to sweep through arbitrarily on this S^2-sphere. When the system pass through these zero-thrust lines, some of its rotors would not be properly functioning, possibly resulting in deterioration of the performance, loss of full-actuation on SE(3)

and even unstable behavior as stated above. Due to this reason, given a task, we call the zero-thrust points on SO(3) of the ODAR system as "ESC-induced singularity".

Now, for this ESC-induced singularity, we propose a novel *selective mapping* algorithm, which, by exploiting the actuation redundancy of the ODAR system, can "propel" at least six rotors far from this singularity and map the desired control wrench $U \in \Re^6$ of (2.12) to these six rotors to maintain the full-actuation on SE(3), while deactivating the control mapping to the (at most) two rotors, that are allowed to be close to the zero-thrust point by the algorithm with some interpolation to enhance the smoothness of this process. Here, we choose to propel only the six rotors away from the singularity rather than all the eight rotors, since: 1) pushing all the eight rotors away from the zero-thrust point necessitates more thrust generation of all the rotors, which turns out too large to accommodate by our ODAR system with its rotor thrust-generation margin already so tight; and 2) six-rotors can still provide the full-actuation on SE(3).

More precisely, for the modified eight-rotor ODAR system as stated in Sec. 2.4.1, we first modulate λ_α of (2.12) such that the thrust magnitude of at least three rotor pairs is larger than a certain thrust margin $\epsilon_1 > 0$ from the zero-thrust line:

$$\lambda_\beta := B^\dagger U + N_B \left(\xi' + \xi''(\lambda_\alpha)\right) \tag{2.15}$$

$$\xi''_\star(\lambda_\alpha) = \arg\min_{\xi''_\star \in \Re} |\xi''_\star - \xi''_{\star,\mathrm{pre}}|$$

$$\text{subj. to} \sum_{i=1}^{4} \mathrm{sgn}\left(|\lambda_{o,\star,i} + \nu_i(\xi'_\star + \xi''_\star)| - \epsilon_1\right) \geq 3 \tag{2.16}$$

where $\star \in \{r, l\}$, $\xi''(\lambda_\alpha) := [\xi''_r; \xi''_l] \in \Re^2$, $\xi''_{\star,\mathrm{pre}} \in \Re$ is the solution from the previous step, and the expression (2.16) comes from the structure of (2.14) with $\nu = [1; -1; -1; 1]$. Here, note that the modulation of ξ'' is done along the column space of the null matrix N_B, which has the form of (2.13) with $\nu = [1; -1; -1; 1]$ for the modified ODAR system of Sec. 2.4.1. This then implies that the optimization of (2.15) is always feasible, since, by increasing $|\xi''_\star|$, we can always "propel" $\lambda_{\alpha,\star}$ along

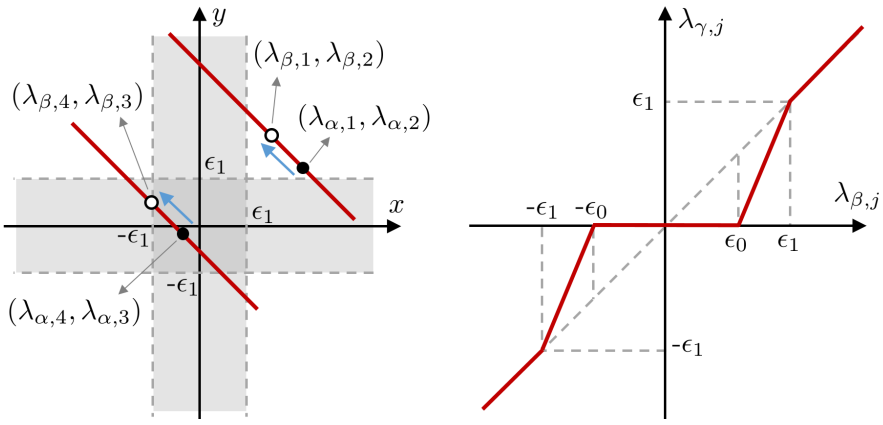

Figure 2-5: (Left) Geometric structure of the λ_β-modulation (2.15): the r-side $\lambda_{\beta,r} = (\lambda_{\beta,1}, \lambda_{\beta,2}, \lambda_{\beta,3}, \lambda_{\beta,4})$ is propelled from the black dots along $\nu_r = [1; -1]$ and $\nu_l = [-1; 1]$ so that only $|\lambda_{\beta,3}| < \epsilon_1$ with the other three larger than ϵ_1; (Right) Thrust thresholds $0 < \epsilon_0 < \epsilon_1$ to gradually switch from the full-use if $|\lambda_{\beta,j}| > \epsilon_1$ to the complete-disuse if $|\lambda_{\beta,j}| < \epsilon_0$ with a linear interpolation between them.

the lines specified by $\nu_r = [1; -1]$ and $\nu_l = [-1; 1]$ outside the set $|\lambda_{\alpha,\star,i}| < \epsilon_1$. See Fig. 2-5 for an illustration of this. This optimization (2.15) can be quickly solved, since the objective function and the constraint are all based on linear functions.

Once the λ_β-modulation of (2.15) is performed, for each r-side and l-side of the ODAR system, we have at most one rotor with its thrust magnitude less than ϵ_1, while the other (at least) three rotors guaranteed to be away from the zero-thrust point farther than ϵ_1. To avoid the ESC-induced singularity, it is then better to stop using those rotors with near-zero thrust, yet, it is not desirable either to suddenly stop using them in view of control smoothness. For this, we define ϵ_0 with $0 < \epsilon_0 < \epsilon_1$ to gradually switch from the full-utilization of the j-th rotor if $|\lambda_{\beta,j}| \geq \epsilon_1$ to its complete-disuse if $|\lambda_{\beta,j}| < \epsilon_0$, with a smooth interpolation between them. See Fig. 2-5.

For the modified eight-rotor ODAR system as stated in Sec. 2.4.1, this gradually-switching selective mapping can be written as follows. For this, suppose first that the j-th and k-th rotor, respectively in the r-side and the l-side of the ODAR system in Fig. 2-1, are designated as the "near-zero" rotors via (2.15), that is, $|\lambda_{\beta,j}| < \epsilon_1$ and $|\lambda_{\beta,k}| < \epsilon_1$ with the thrust magnitude of all the other six rotors larger or equal

to ϵ_1. Here, we allow $j = \emptyset$ or $k = \emptyset$. Then, we further modulate λ_β of (2.15) s.t.,

$$\lambda_\gamma := \bar{B}^\dagger(\lambda_\beta)U + \bar{N}_B(\lambda_\beta)(\xi' + \xi''(\lambda_\alpha)) \tag{2.17}$$

with

$$\bar{B}^\dagger(\lambda_\beta) := \begin{bmatrix} e_r B_r^\dagger + (1-e_r)B_{r\backslash\{j,k\}}^\dagger \\ e_l B_l^\dagger + (1-e_l)B_{l\backslash\{j,k\}}^\dagger \end{bmatrix} \in \Re^{8\times 6}$$

$$\bar{N}_B(\lambda_\beta) := \begin{bmatrix} e_r\nu & 0 \\ 0 & e_l\nu \end{bmatrix} \in \Re^{8\times 2}$$

and

$$e_\star := \begin{cases} 1 & \text{if } |\min_{i\in\mathcal{N}_h}(|\lambda_{\beta,\star,i}|)| > \epsilon_1 \\ \frac{\min_{i\in\mathcal{N}_h}(|\lambda_{\beta,\star,i}|)-\epsilon_0}{\epsilon_1-\epsilon_0} & \text{if } \epsilon_1 \geq |\min_{i\in\mathcal{N}_h}(|\lambda_{\beta,\star,i}|)| > \epsilon_0 \\ 0 & \text{if } \epsilon_0 \geq |\min_{i\in\mathcal{N}_h}(|\lambda_{\beta,\star,i}|)| \end{cases}$$

for $\star \in \{r, l\}$, where $B_r^\dagger, B_l^\dagger \in \Re^{4\times 6}$ are defined by $[B_r^\dagger; B_l^\dagger] := B^\dagger \in \Re^{8\times 6}$, and $B_{r\backslash\{j,k\}}^\dagger, B_{l\backslash\{j,k\}}^\dagger \in \Re^{4\times 6}$ by $B_{\backslash\{j,k\}}^\dagger =: \left[B_{r\backslash\{j,k\}}^\dagger; B_{l\backslash\{j,k\}}^\dagger\right] \in \Re^{8\times 6}$ where $B_{\backslash\{j,k\}}^\dagger := I_{\backslash\{j,k\}}B^T(BI_{\backslash\{j,k\}}B^T)^{-1}$ is the reduced mapping matrix excluding the j-th and k-th rotors with $I_{\backslash\{j,k\}} \in \Re^{8\times 8}$ being the identity matrix with the j-th and k-th diagonal elements set to be zero. Here, $(BI_{\backslash\{j,k\}}B^T)^{-1}$ is non-singular from the structure of $B_r, B_l \in \Re^{3\times 4}$ and $BB_{\backslash\{j,k\}}^\dagger = I$.

When the properties of (2.11) are granted, as true for the modified eight-rotor ODAR system explained in Sec. 2.4.1, we then have the following "decoupling" property, i.e.,

$$B_{r\backslash\{j,k\}}^\dagger = B_{r\backslash j}^\dagger, \quad B_{l\backslash\{j,k\}}^\dagger = B_{l\backslash k}^\dagger \tag{2.18}$$

where $B^{\dagger}_{r\backslash j}, B^{\dagger}_{l\backslash k} \in \Re^{4\times 6}$ are defined by

$$B^{\dagger}_{\backslash j} =: \left[B^{\dagger}_{r\backslash j}; B^{\dagger}_{l}\right], \quad B^{\dagger}_{\backslash k} =: \left[B^{\dagger}_{r}; B^{\dagger}_{l\backslash k}\right] \tag{2.19}$$

where $B^{\dagger}_{\backslash i} := I_{\backslash i} B^T (B I_{\backslash i} B^T)^{-1}$. Here, if $j = \emptyset$ or $k = \emptyset$, $B^{\dagger}_{r\backslash j} = B^{\dagger}_{r}$ (with $e_r = 1$) or $B^{\dagger}_{l\backslash k} = B^{\dagger}_{l}$ (with $e_l = 1$). This then means that the selective mapping matrix $B^{\dagger}_{\star\backslash\{j,k\}}$ for (2.17) can be computed for the r-side (i.e., $B^{\dagger}_{r\backslash j}$) and the l-side (i.e., $B^{\dagger}_{l\backslash k}$) as if they are decoupled from each other. This decoupling property (2.18)-(2.19) also turns out crucial to render the gradually-switching selective mapping (2.17) to be exact (i.e., produce the desired control wrench U regardless of e_r, e_l), as summarized in the next Prop. 2.

Proposition 2. *Consider the mapping matrix $B \in \Re^{6\times 8}$ of* (2.8) *for the eight-rotor ODAR system. Then, if the properties of* (2.11) *are satisfied:*

1. *The decoupling property* (2.18)-(2.19) *is granted; and*

2. *The mapping* (2.17) *is exact, i.e., $B\lambda_{\gamma} = U$, $\forall e_r, e_l$.*

Proof. Write $B^{\dagger} = [B^{\dagger}_{r,f}\ B^{\dagger}_{r,\tau}; B^{\dagger}_{l,f}\ B^{\dagger}_{l,\tau}]$, where $B^{\dagger}_{\star,*} \in \Re^{4\times 3}$, $\star \in \{r, l\}$ and $* \in \{f, \tau\}$. By using the Sherman-Morrison formula, $B^{\dagger}_{\backslash\{j,k\}}$ in (2.18)-(2.19) can be expanded s.t.,

$$\begin{aligned} B^{\dagger}_{\backslash\{j,k\}} &= I_{\backslash\{j,k\}} B^T (B I_{\backslash\{j,k\}} B^T)^{-1} \\ &= I_{\backslash\{j,k\}} \left(B^{\dagger} - B^{\dagger} B I_{\{j,k\}} S^{-1} I_{\{j,k\}} B^{\dagger}\right) \end{aligned} \tag{2.20}$$

where $I_{\{j,k\}} := I_{8\times 8} - I_{\backslash\{j,k\}} \in \Re^{8\times 8}$ and $S := I_{8\times 8} + I_{\{j,k\}} B^{\dagger} B I_{\{j,k\}}$.

Here, we first show that $B^{\dagger}B$ is block diagonal. For this, we write

$$B^{\dagger}B = \frac{1}{2}\begin{bmatrix} B^{\dagger}_{\star,f}B_{f,\star} + B^{\dagger}_{\star,\tau}B_{\tau,\star} & B^{\dagger}_{\star,f}B_{f,\star} - B^{\dagger}_{\star,\tau}B_{\tau,\star} \\ B^{\dagger}_{\star,f}B_{f,\star} - B^{\dagger}_{\star,\tau}B_{\tau,\star} & B^{\dagger}_{\star,f}B_{f,\star} + B^{\dagger}_{\star,\tau}B_{\tau,\star} \end{bmatrix}$$

due to the property of (2.11), where $\star \in \{r, l\}$. Further, since they share the same null-vector $\nu \in \Re^4$ in (2.11), we can write

$$B_{f,\star} = B_{f,\star\backslash a} L, \quad B_{\tau,\star} = B_{\tau,\star\backslash a} L$$

where $a \in \mathcal{N}_h$ is an index s.t., $\nu_a \neq 0$, $B_{f,\star\backslash a}, B_{\tau,\star\backslash a} \in \Re^{3\times 3}$ are matrices obtained by discarding the a-th column from $B_{f,\star}, B_{\tau,\star}$, and $L \in \Re^{3\times 4}$ is the constant matrix specified by ν of (2.11). Here, since $B_{f,\star}, B_{\tau,\star}$ are designed to be full row-rank (i.e., 3) for the full-actuation on SE(3), $B_{f,\star\backslash a}$ and $B_{\tau,\star\backslash a}$ are also full rank and invertible. We can then obtain $B^\dagger_{\star,f} B_{f,\star} = B^\dagger_{\star,\tau} B_{\tau,\star} = = L^T (L L^T)^{-1} L$, and further

$$B^\dagger B = \mathrm{diag}[L^T (L L^T)^{-1} L, L^T (L L^T)^{-1} L]$$

Then, since $B^\dagger B$ is block diagonal and also only the j-th and k-th rows of $I_{\{j,k\}} S^{-1} I_{\{j,k\}} B^\dagger$ are non-zero, the upper four rows of $B^\dagger_{\backslash\{j,k\}}$ in (2.20) are the same as those of $B^\dagger_{\backslash j}$, whereas its lower four rows the same as those with $B^\dagger_{\backslash k}$. If we set $k = \emptyset$ or $j = \emptyset$, $B^\dagger_{\backslash\{j,k\}} = B^\dagger_{\backslash j}$ or $B^\dagger_{\backslash\{j,k\}} = B^\dagger_{\backslash k}$, which still retain the same structure of $B^\dagger_{\backslash\{j,k\}}$. This complete the proof of the first item.

For the second item, recall that $BB^\dagger = B_r B_r^\dagger + B_l B_l^\dagger = I$. Also, from the decoupling property (2.18)-(2.19), we have

$$B B^\dagger_{\backslash j} = B_r B^\dagger_{r\backslash j} + B_l B_l^\dagger = I$$

where $B^\dagger_{\backslash j} = I_{\backslash j} B^T (B I_{\backslash j} B^T)^{-1}$ with $(B I_{\backslash j} B^T)^{-1}$ being invertible from the structure of B_r, B_l as stated above for $B^\dagger_{\backslash\{j,k\}}$. This, and the similar derivation for $BB^\dagger_{\backslash k}$, imply that

$$B_r B_r^\dagger = B_r B^\dagger_{r\backslash j}, \quad B_l B_l^\dagger = B_l B^\dagger_{l\backslash k}$$

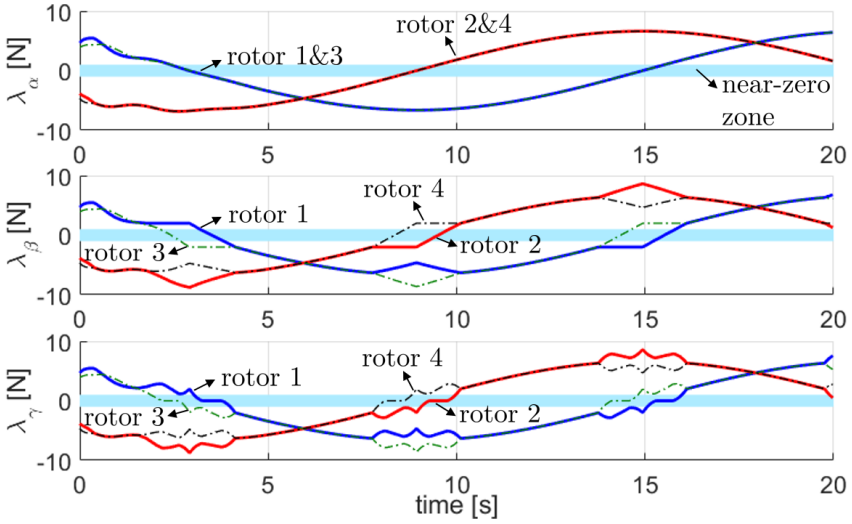

Figure 2-6: Selective mapping process: $\lambda_{\alpha,i}$ (thrust value after infinity-norm minimization), $\lambda_{\beta,i}$ (thrust value after full-actuation preserving modulation) and $\lambda_{\gamma,i}$ (thrust value after excluding zero-crossing rotors) during the pitching rotation motion. Only $\lambda_{\star,i}$ of the r-side rotors are shown for brevity.

with which we have

$$\begin{aligned} B\bar{B}^\dagger &= e_r B_r B_r^\dagger + (1-e_r) B_r B_{r\backslash j}^\dagger + e_l B_l B_l^\dagger + (1-e_l) B_l B_{l\backslash k}^\dagger \\ &= I + e_r (B_r B_r^\dagger - B_r B_{r\backslash j}^\dagger) + e_l (B_l B_l^\dagger - B_l B_{l\backslash k}^\dagger) = I \end{aligned}$$

completing the proof of the second item. □

The infinity-norm optimization (2.12) and the selective mapping process (2.15) and (2.17) are shown in Fig. 2-6, where $\lambda_{\alpha,i}$, $\lambda_{\beta,i}$ and $\lambda_{\gamma,i}$ are plotted during a pitching rotation simulation of the modified eight-rotor ODAR system, which in fact passes through the intersection of the two zero-thrust lines in Fig. 2-4. Only those for the r-side rotors are shown here with that of the l-side omitted for brevity. From Fig. 2-6, we can then see that: 1) $(\lambda_{\beta,1}, \lambda_{\beta,3})$ split around 3sec and 14sec, whereas $(\lambda_{\beta,2}, \lambda_{\beta,4})$ around 8sec, preventing simultaneous zero-crossing of multiple rotors and allowing only for one rotor thrust magnitude to be less than ϵ_1, while that of the other three larger than ϵ_1, thereby, ensuring the full-actuation on SE(3); and

2) through the process of (2.17), the $\lambda_{\gamma,i}$ behaves around near-zero boundary more smoothly than $\lambda_{\beta,i}$, while the rotors with near-zero thrust are gradually switched to disuse and, when $|\lambda_{\beta,i}| < \epsilon_o$, their thrust is set to zero according to (2.17) (e.g., $\lambda_{\gamma,1} = 0$ around 3sec and 16sec). See also Fig. 2-14 for the experimental result of this selective mapping.

We also perform the simulation of hovering with the fixed 45° pitch angle, which causes four rotors to be simultaneously near zero-thrust. For this, we model the ESC-induced singularity s.t., given the command thrust λ_i^d, the real thrust output λ_i is given by

$$\dot{\lambda}_i = \begin{cases} k(0 - \lambda_i) & (\lambda_i^d \lambda_i < 0 \ \text{ and } \ |\lambda_i| > \epsilon) \\ k(\lambda_i^d - \lambda_i) & (\text{otherwise}) \end{cases}$$

so that, when the rotor is commanded to reverse its rotating direction, it approaches to zero-thrust first, then, re-accelerate after staying there for some time specified by $k > 0$. We use the same parameters and control gains, yet, one with the selective mapping while the other not. The results shown in Fig. 2-7, where we can clearly see that, without the selective mapping, the ODAR system behavior becomes fairly shaky with λ_i swaying back and forth from λ_i^d whenever it crosses zero-thrust, and, consequently, cannot maintain the hovering, whereas, with the selective mapping, it can attain the hovering with the full-actuation in SE(3) guaranteed while also substantially subduing the frequency of (severely performance-degrading) zero-crossing as compared to the case of the no selective mapping.

It is yet worthwhile to mention that, even if it is exact as proved in Prop. 2 and also its efficacy manifested in Fig. 2-7, the selective mapping, in general, can only alleviate the ESC-induced singularity, not completely eliminate it. This is because the selective mapping formulates "dynamic" ESC-induced singularity as "static" entities (i.e., conditions on thrust magnitude). Due to this reason, the ODAR system should be designed in such a way that: 1) the target operations mostly

take place as far from the ESC-induced singularity as possible; and 2) crossing zero-thrust lines is permitted with the selective mapping, yet only in a reserved manner. Open-loop control with a proper motion planning may also be used together with the (closed-loop) selective mapping, although the behavior attainable by this open-loop approach complying with the ESC-induced singularity is rather limited and its analysis typically complicated (e.g., fast rotation quickly passing through singularity [20]). How to incorporate the ESC-induced singularity with its full dynamics and nonlinearity is a topic of active research by itself (i.e., control optimization with dynamic constraint) and is a topic of our future research as well.

2.5 Experiment

2.5.1 System Setup

As stated in Sec. 2.2.2 and Sec. 2.4.1, we implement the modified eight-rotor ODAR as shown in Fig. 2-1, whose main bar-frame is constructed by using a commercial carbon fiber pipe with 20mm diameter and 1.5mm thickness, making the length of the total system to be 1.2m. Eight BLDC motors (MN3508-KV700 from T-Motor) are attached to the mainframe via 3D-printed parts according to the design optimization of Sec. 2.2.2. To achieve bi-directional rotors, we stack two uni-directional props (each with two blades, diameter 10", pitch 4.7") in the opposite direction as shown in Fig. 2-8 and drive them together by a reversible ESC (DYS-XMS30A with BLHeli firmware). We also perform an experiment to check the thrust generation of this bi-directional rotor and achieve the result as in Fig. 2-9, which shows that this bi-directional rotor can produce thrust about 92% of that of the single prop, which is up to 9.7 N for both the upward and downward directions. To provide enough current all to the eight rotors, we adopt a 4-cell Li-Po battery (14.8V, 4000mAh, 45C) along with SBEC (switching battery eliminator circuit) to power up other modules (e.g., computing, communication, etc.) with 5V.

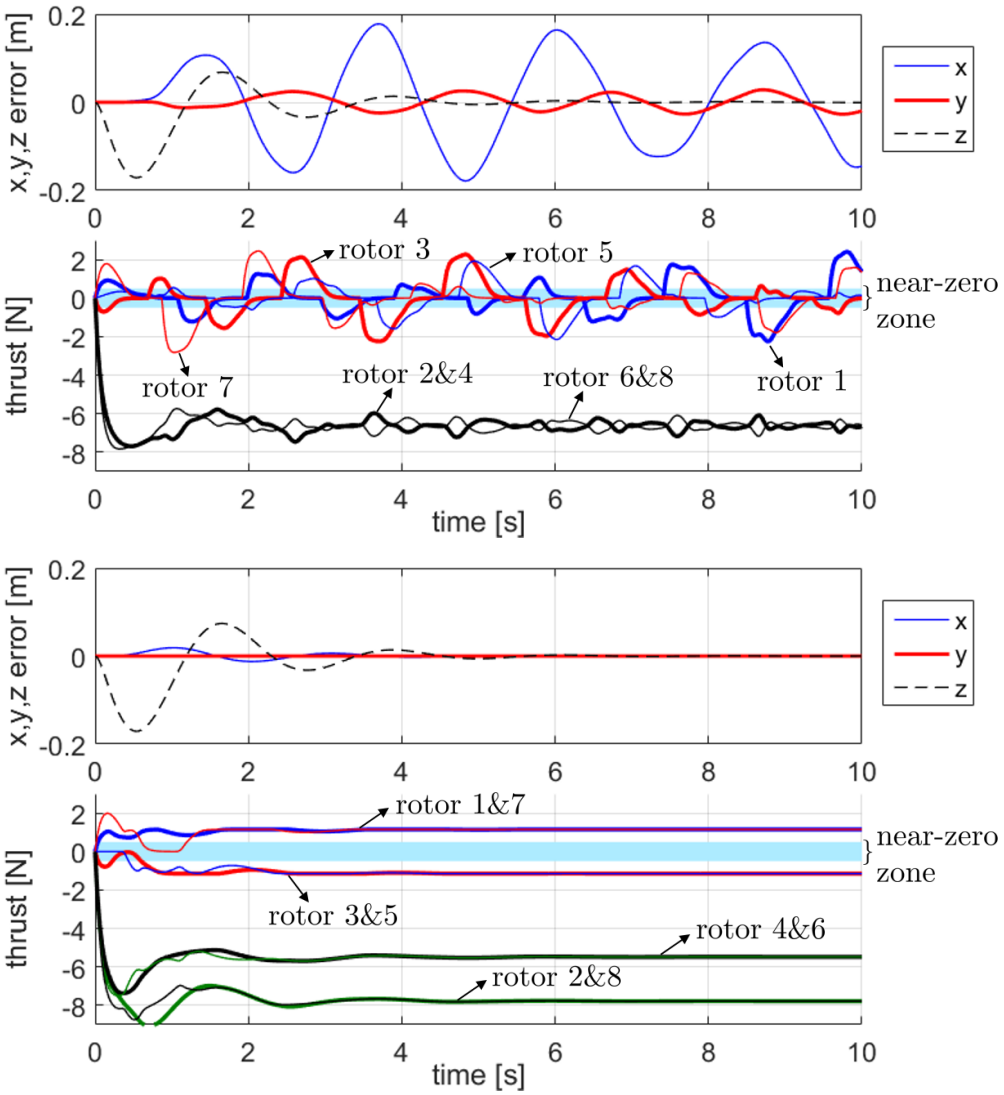

Figure 2-7: Simulation of the ODAR system hovering with fixed pitch angle 45° without the selective mapping (top) and with the selective mapping (bottom).

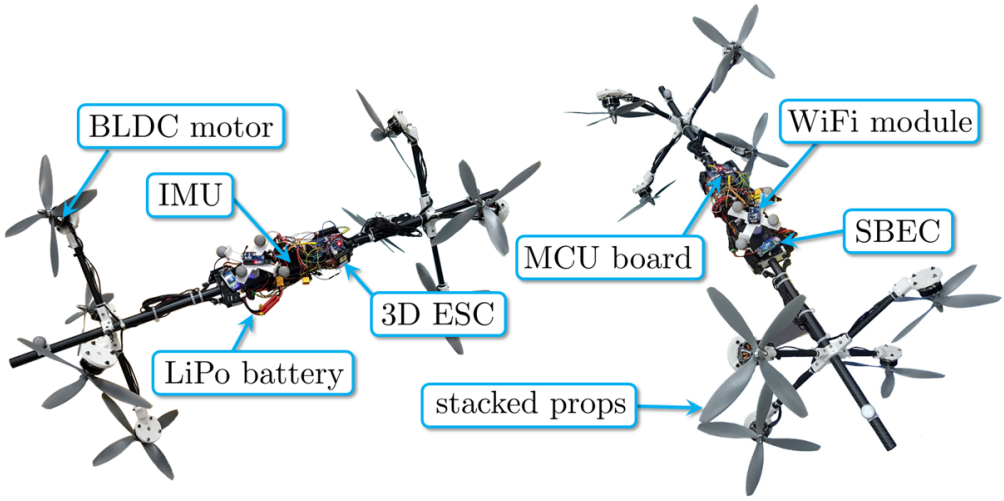

Figure 2-8: Prototype of the eight-rotor ODAR system and description of each component

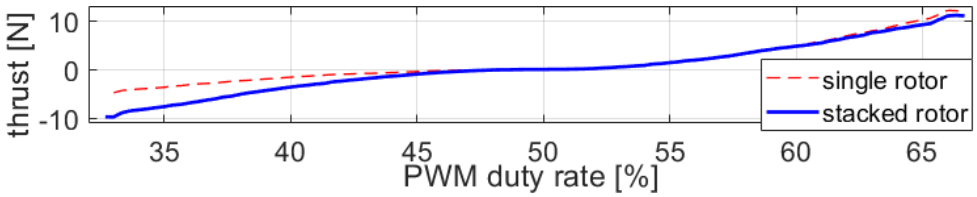

Figure 2-9: Thrust generation of rotor with one single uni-directional prop (dashed line) and with two uni-directional props stacked in opposite direction (solid line).

For the computing, an MCU (micro controller unit) board equipped with Cortex-M4 CPU (STM32F429IG from STMicroelectronics) is used. The central vacancy of the mainframe contains these battery, MCU and other modules to make the geometric center coincident to the mass center as close as possible. With all these, the final ODAR system is achieved as shown in Fig. 2-1 with the weight of 2.6kg. See also Sec. 2.2.2 for the other system specifications.

The MCU board then receives the pose data from a motion capture system (VICON Bonita-B3) via Wi-Fi (2.4GHz) with 125Hz, which is fused with the gyroscope measurement of an IMU sensor (MPU-9250) with 200 Hz via I^2C to obtain the attitude of the ODAR system through the SO(3) nonlinear complementary filter [37]. The desired pose for the control is computed in the MCU board, or, in the case of teleoperation, received from a 6-DOF haptic device (Phantom Premium 6DOF) via Wi-Fi with 125Hz. External wrench is estimated with the wrench estimator [38] and also measured with a 6-axis F/T sensor (RFT40-SA01, Robotous) as ground truth for the hybrid pose/wrench control. For teleoperation, we use the F/T sensor to avoid the implementation issue of communication (i.e., sending estimated wrench from MCU to the master device significantly slows down Wi-Fi from MOCAP to MCU as well). With all the information as stated above, the MCU board calculates the desired control wrench (f, τ) as stated in Sec. 2.3 and allocates it to the eight rotors via the infinity-norm optimization (Sec. 2.4.1) modulated by the selective mapping (Sec. 2.4.2) with 1kHz, which is then converted to PWM (pulse width modulation) signal and sent to each reversible ESC.

2.5.2 Experiment Results

Using the ODAR system constructed as in Sec. 2.5.1, we conduct the validating experiments of the three control laws of Sec. 2.3: 1) pose trajectory tracking; 2) hybrid pose/wrench control; and 3) PSPM-based peg-in-hole teleoperation. We also perform an experiment to show the efficacy of the selective mapping of Sec. 2.4.2

for the pitching rotation with four rotors experiencing the ESC-induced singularity. Due to the page limit, here, we only present the (partial) results of the experiments mentioned above, compressively showing the indispensable results to explain the contributed works. We then refer readers to the accompanied video for full experimental results[1] including results of pose trajectory tracking, hybrid control and peg-in-hole teleoperation with other attitude to thoroughly demonstrate the performance of the system.

First of all, motion control in SE(3) designed in Sec. 2.3.2 is evaluated by circular trajectory tracking with horizontal/vertical (pitch angle 0° and 90°) attitude. A trajectory of a circle with 1.0 m diameter is given and the system is required to track the trajectory with its center of mass and rotate with 36°/s maintaining the attitude to be $[\varphi_d; \theta_d; \psi_d] = [0; 0; 0]$ [deg] and $[\varphi_d; \theta_d; \psi_d] = [0; 90; 0]$ [deg] where $\varphi_d, \theta_d, \psi_d$ are desired roll, pitch, and yaw angle in $\{O\}$. The results of the experiment are shown in Fig. 2-10. The RMS error with horizontal attitude is 6.69 cm, 3.04° for position and attitude respectively, and 5.83 cm, 2.03° with vertical attitude. The results show that the position and attitude of the ODAR system can be controlled simultaneously with small enough error using full-DOF actuation.

Then, the results of the hybrid pose/wrench control are shown in Fig. 2-11, where the modified eight-rotor ODAR system tracks the circular trajectory of 60cm diameter with the angle-sweeping rate of 9°/s, while maintaining the vertical attitude and pushing down the horizontal board with 10N. The RMS errors of the position, angle and force tracking are obtained to be 3.04cm, 1.01° and 0.82N. For this, we can clearly see that the ODAR system is fully-actuated on SE(3). We also find that the system can fairly stably interact with the board even with substantial ground-flow effect, which we believe is due to its bar-shape.

For more accurate assessment of the wrench control performance in constrained space designed in Sec. 2.3.3, supplementary experiments are done along with force/-torque measurement. First, an experiment of checking the external wrench estimator

[1] Also, available at `https://youtu.be/S3i9NspWtr0`.

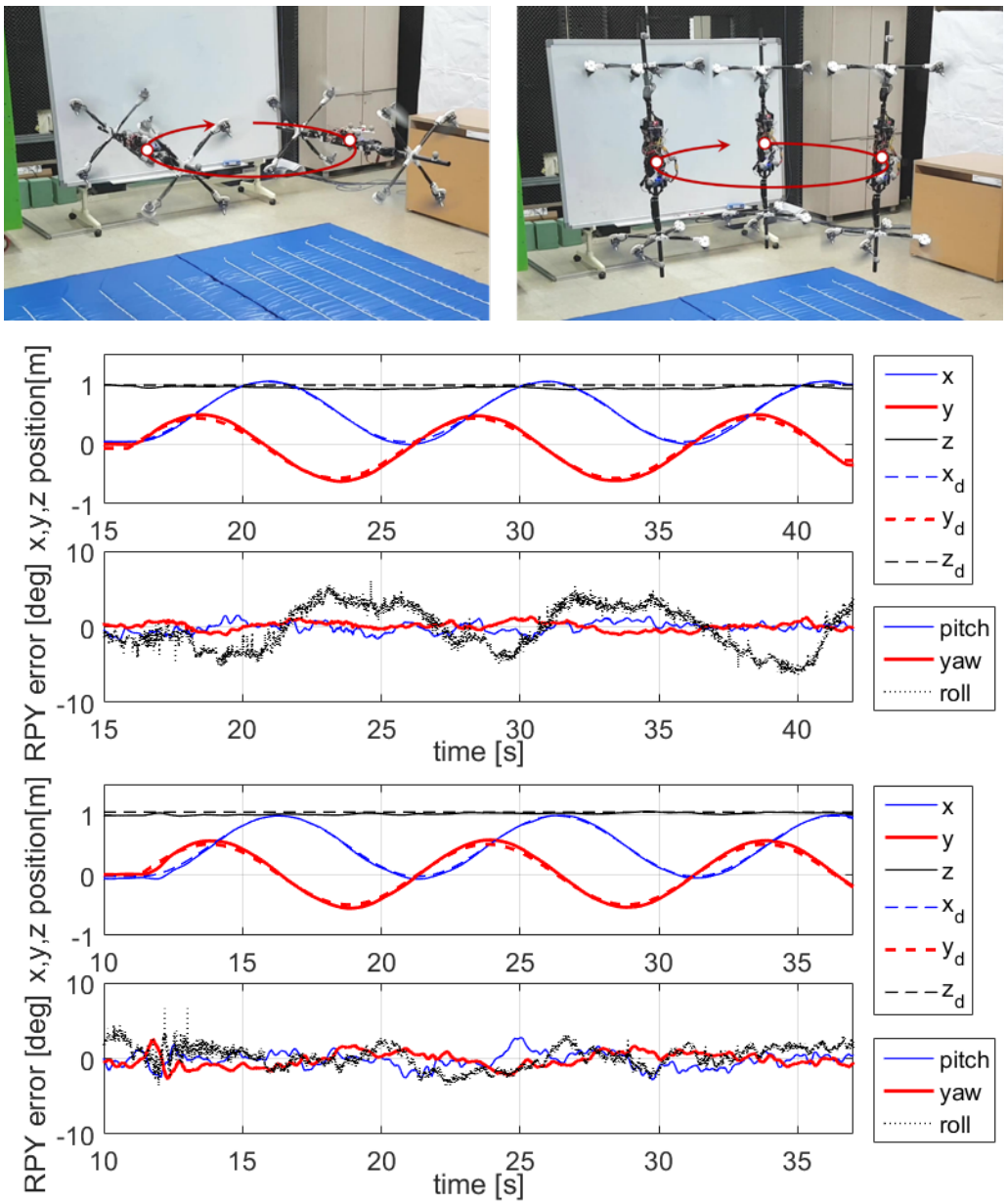

Figure 2-10: Circular trajectory tracking: position and attitude error while maintaining the attitude to be (1st & 2nd) $[\varphi_d; \theta_d; \psi_d] = [0; 0; 0]$ [deg], and (3rd & 4th) $[\varphi_d; \theta_d; \psi_d] = [0; 90; 0]$ [deg] in euler angles.

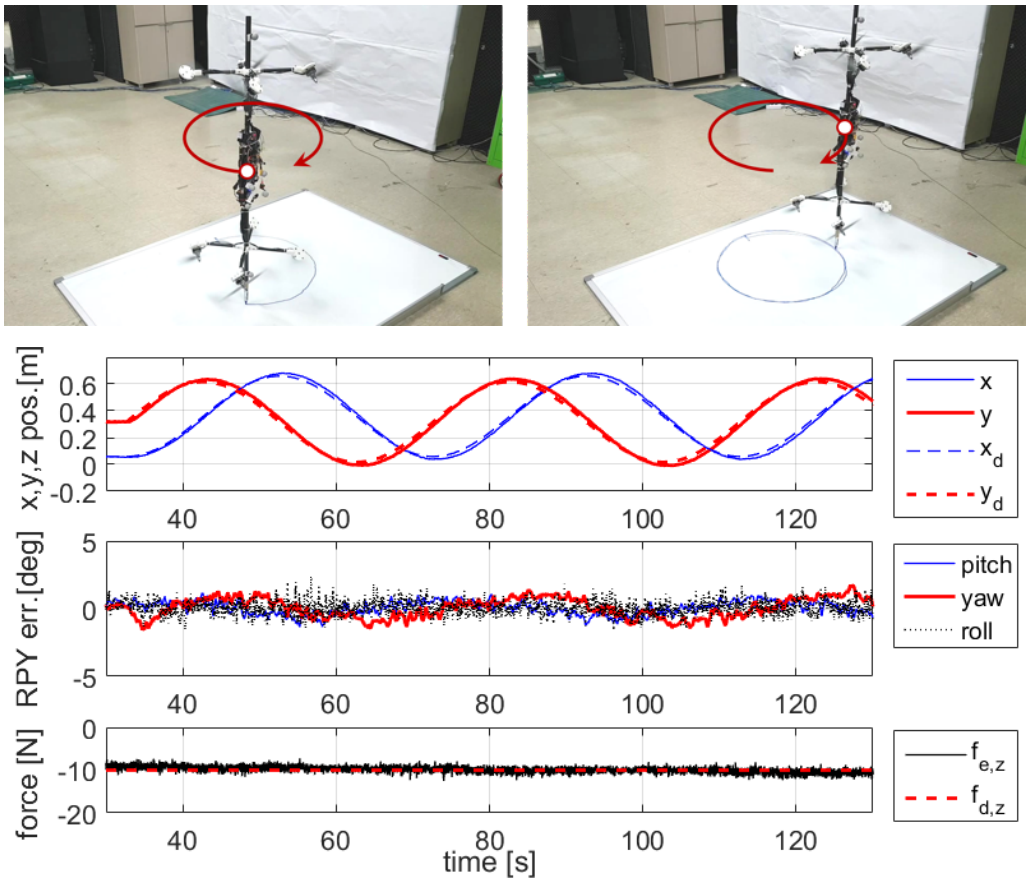

Figure 2-11: Hybrid pose/wrench control: pose tracking error and contact force regulation performance ($f_{e,z}$, $f_{d,z}$: measured and desired applying force in z-direction) while drawing the circle on the horizontal plane.

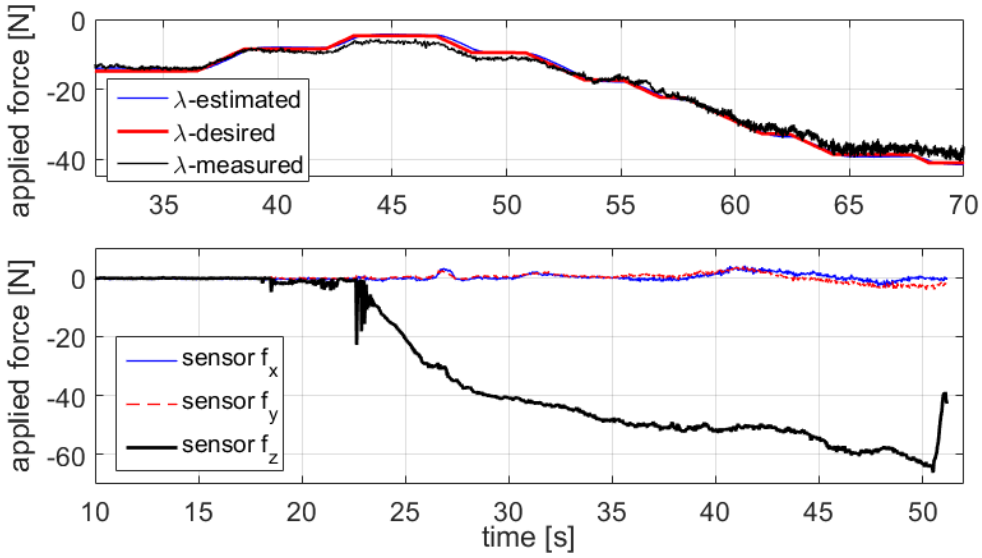

Figure 2-12: (Top) Estimated, desired and measured contact force ($\hat{\lambda}_c$, $\lambda_{c,d}$, λ_c) during the downward pushing on the ground; (Bottom) Measured contact force while strongly pushing down with vertical attitude.

and wrench control is done. In the experiment, the system is required to push downward on the F/T sensor with vertical attitude ($[\varphi_d; \theta_d; \psi_d] = [0; 90; 0]$ [deg]) while controlling the contact wrench so that the wrench can be measured and compared with the desired, estimated one. As you can see from the plot in the top of the Fig. 2-12, the desired, estimated and measured contact force are almost coinciding within the force range of 5 N to 40 N. Though, there exists a slight error when the contact force is far from the 25 N which is the weight of the system itself, and this error is supposed to be caused by the model uncertainty of the actuator. Also, the similar experiment is done again to check the manipulation capability of the ODAR system, by strongly pushing down on the F/T sensor with vertical attitude. The result of sensor measurement is demonstrated in the bottom of Fig. 2-12, and the system turned out to be able to push down upto 64 N (39 N excluding weight of the system) which is large enough manipulation force much larger than its own weight, impossible for common uni-directional rotor-based aerial manipulation systems.

The results of the PSPM-based peg-in-hole teleoperation are shown in Fig. 2-13, where the ODAR system is teleoperated by a human user in such a way that the

end of its carbon-fiber tube with the diameter of 20mm is inserted into a 3D-printed hole of the diameter of 21mm. This level of task precision (i.e., radial tolerance of 0.5mm) is by far beyond the achievable by some fully autonomous control as demonstrated in the above hybrid control experiment (i.e., RMS positioning error of 3.04cm). This we believe is because the system dynamics with rotor aerodynamics, fluid-structure interaction, motor dynamics, unmodeled structural dynamics, etc. is too complicated to be modeled in a mathematically tractable way. Even so, with the teleoperation, the human user can succeed this precision peg-in-hole task from the first contact (around 22sec) to its full insertion (around 36sec). This we believe is because human users somehow can "learn" those complicated physics and incorporate that into their commanding strategy to properly react to those complicated physics of the system, thereby, successfully guiding its dynamics into the peg-in-hole completion. For this, we also find the 3D visual information and the force feedback are imperative, without which the human users find it very difficult to achieve this task.

The last experiment is to show the efficacy of the selective mapping in Sec. 2.4.2 to subdue the instability stemming from the ESC-induced singularity. For this, the ODAR system is controlled to rotate from 90° pitch angle (i.e., vertical posture) to 0° pitch angle (i.e., horizontal posture), while maintaining its center-of-mass position and other attitude angles stationary, inducing the zero-thrust crossing of four rotors at the same time as shown in Fig. 2-4. The experiment results in Fig. 2-14 show that: 1) without the selective mapping, the behavior of the ODAR systems becomes so shaky (e.g., wobbling around 17sec and 21sec), resulting in the failure of the pitching rotation experiment (eventually unstable crash if not stopped by hands); and 2) with the selective mapping, this shaky and unstable behavior is successfully subdued and the ODAR system can finish the vertical-to-horizontal pitching rotation.

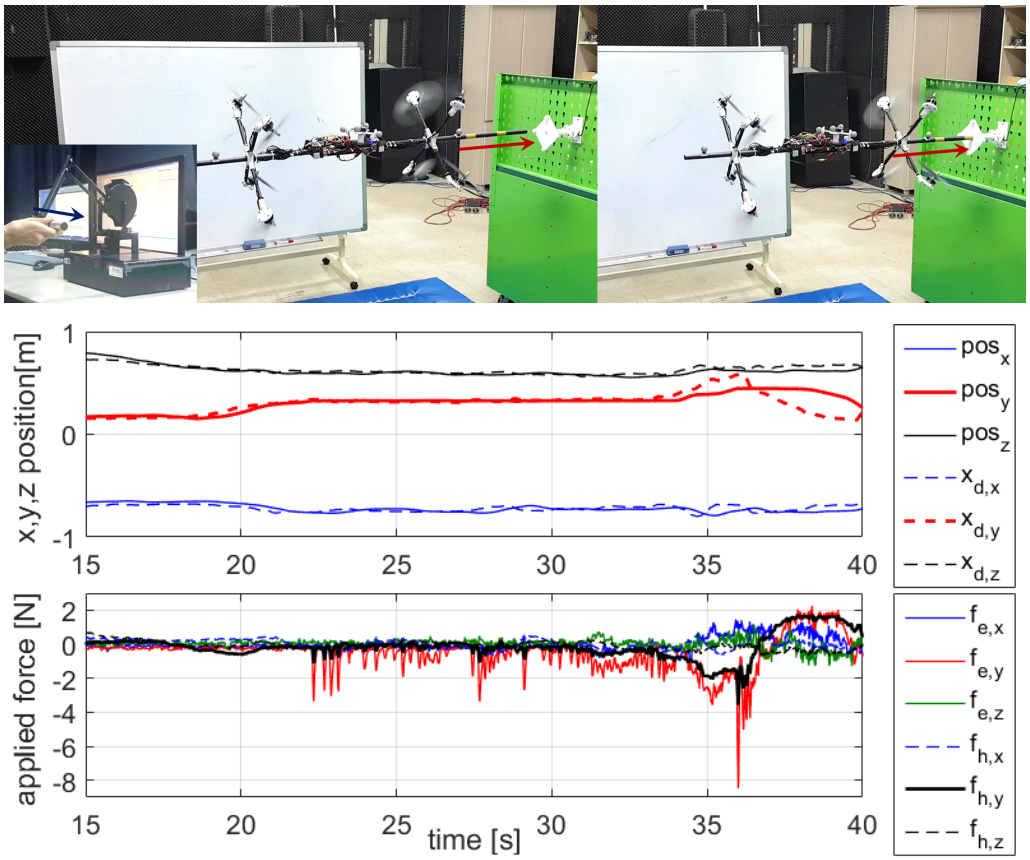

Figure 2-13: Peg-in-hole teleoperation: (Top) system position x and human command x_d; (Bottom) measured external force f_e and haptic force feedback f_h.

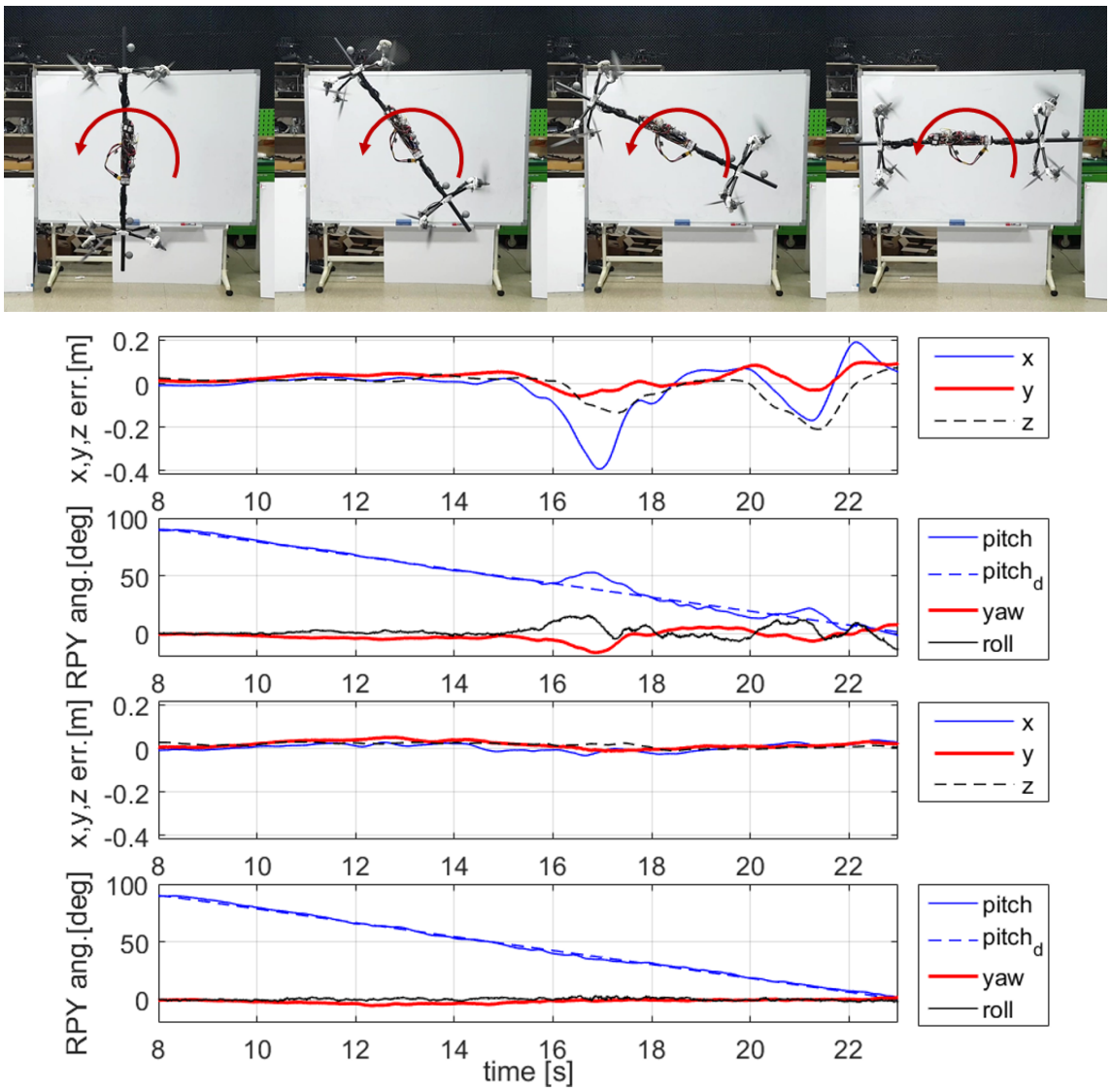

Figure 2-14: Pitching-rotation with the center-of-mass position and other attitudes hold: without selective mapping (top two plots) and with selective mapping (bottom two plots).

2.6 Conclusion

In this chapter, we propose a new aerial manipulation platform, ODAR (omni-directional aerial robot), which can produce omni-directional motion and wrench (i.e., full-actuation in SE(3)). To address the tight thrust margin and weight budget of current motor and battery technologies, we present a design optimization framework, which incorporate such important aspects as inter-rotor aero-interference, anisotropic task requirement, etc. Closed-form infinity-norm optimal control allocation and selective mapping algorithm are also proposed to address the tight thrust saturation margin and the ESC-induced singularity of sensor-less BLDC rotors. With all these, the ODAR system exhibits the following unprecedented level of performance and capability: 1) separate position and orientation control on SE(3); 2) hybrid pose/wrench control with downward force of 60N much larger than its own weight (2.6kg); and 3) peg-in-hole force feedback teleoperation with radial tolerance of 0.5mm.

Chapter 3

Pose and Posture Estimation of an Aerial Skeleton System

3.1 Introduction

In this chapter, we consider the onboard pose and posture estimation problem of the LASDRA system [3, 4] particularly for its outdoor flying (see Fig. 3-1). This problem is challenging for the following reason. First, to maximize system dexterity, the LASDRA system connects two links via a cable. This then allows for full 3-DOF inter-link rotation, which is difficult to measure by (accurate, yet, axis-demanding) encoders, and instead, IMUs (inertial measurement units) are more suitable (or often only viable) option for that as so for our LASDRA system. This IMU however exhibits non-negligible (yet, still bounded) link absolute attitude estimation error, which can be accumulated to a rather very large link position estimation error as propagated toward the end of the aerial skeleton system. This can pose a serious problem, since, e.g., if the skeleton is supposed to fly with a certain posture, this desired behavior essentially needs to be decoded into the position and attitude controls of each link, and, with the (accumulated) link position estimation error becoming very large (e.g., skeleton with large number of links), the link position control can

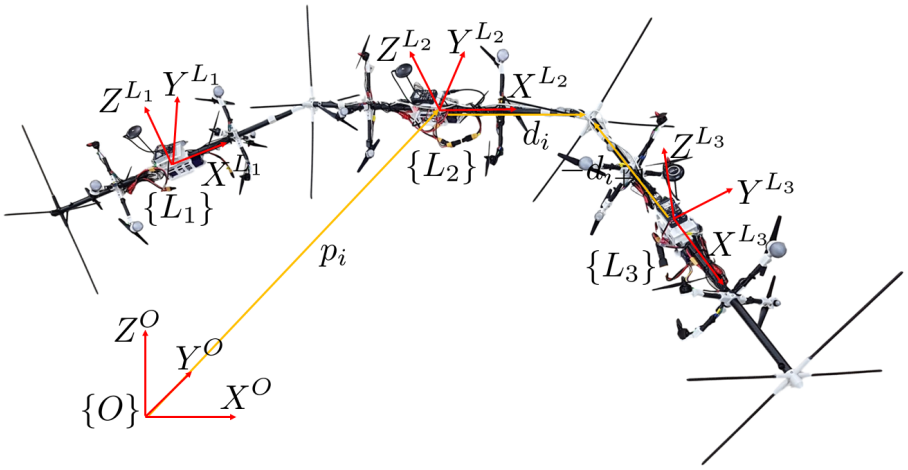

Figure 3-1: Aerial skeleton system: three link LASDRA system for outdoor flying. Also shown are the inertial and link coordinate frames, $\{O\} := \{X^O, Y^O, Z^O\}$, $\{L_i\} := \{X^{L_i}, Y^{L_i}, Z^{L_i}\}$.

be excessively erroneous, which may contradict/conflict with the attitude/position control of its own or other neighboring links and possibly result in collapsing and falling-down of the skeleton (with excessive internal force at some joints) particularly with the typical rotors prone to saturate. We may attach a GPS (global positioning system) module on each link to correct its position estimation error, which is still not so promising either, since the accuracy of typical GPS is too poor (i.e., meter-level accuracy) to reclaim the kinematic coherency of the aerial skeleton system as stated above.

To overcome this challenge, in this thesis, we propose a novel pose and posture estimation framework for the aerial skeleton system based on IMU and GPS sensors for its outdoor flying. More precisely, we attach an IMU sensor and a GPS module on each link and estimate their state via standard SE(3)-motion EKF (extended Kalman filtering) [39], [40]. We choose this (distributed) sensor configuration to render the skeleton system "modular" while also significantly enhancing position sensing accuracy with redundant/independent GPS sensors (via the constraints - see Sec. 3.5). We then apply the kinematic constraints of the aerial skeleton system (i.e., point-constraint between two links via the cable) to these EKF-estimates of all the

links to enforce the kinematic coherency and, consequently, (significantly) improve the estimation accuracy. For this, we adopt the framework of smoothly constrained Kalman filtering (SCKF) [14], where nonlinear constraint is linearized and applied repeatedly as measurement updates with some artificial noise. We choose SCKF here is because it is known of its superior performance of constraint error convergence as compared to other constrained KF techniques (see [41]), which is crucial for this thesis as it can directly translate to the performance of enforcing the kinematic coherency of the skeleton system. We also extend the standard SCKF in this thesis to incorporate the multi-dimensional kinematic constraints, error-state formulation and some suitable manifold structure common in SE(3)-motion estimation. We further devise a scalable semi-distributed version of the estimation algorithm, which can substantially speed up the computation speed by dividing the skeleton into several groups, locally-performing full-SCKF for each group, and globally-performing partial-SCKF among the groups. The presented estimation frameworks are then verified with real outdoor flying experiments and simulation studies of our LASDRA system. To our knowledge, we presents the very first result on the onboard estimation framework of the aerial skeleton system and its real outdoor flying demonstration.

3.2 Preliminary

A general aerial skeleton system can be modelled with Newton-Euler dynamics as following,

$$m_i\ddot{p}_i + m_i g e_3 = R_i u_i + R_i f_i - R_{i+1} f_{i+1}$$
$$J_i\dot{\omega}_i + S(\omega_i)J_i\omega_i = \tau_i + S(r_{ci,i})f_i - S(r_{ci,i+1})R_{i+1}f_{i+1}$$

where $m_i \in \Re, J_i \in \Re^{3\times3}$ are the mass and inertia matrix of i-th link, $p_i, \omega_i \in \Re^3$ are the position and angular velocity vector of i-th link expressed in inertial frame

$\{O\}$ and link frame $\{L_i\}$ respectively (see Fig. 3-1 for the definition of frames), $R_i \in \mathrm{SO}(3)$ is the rotation matrix of i-th link, $u_i, \tau_i \in \Re^3$ are force and torque input applied to i-th link in $\{L_i\}$, $f_i, f_{i+1} \in \Re^3$ are the force applied at left and right side joint of the i-th link (here, right side means positive x direction in link frame $\{L_i\}$), $g \in \Re$ is the gravitational constant with $e_3 := [0; 0; 1]$, and $S(\star)$ is the skew-symmetric matrix mapping.

In this thesis, the LASDRA system [3] is exploited as an aerial skeleton, each link module of which is comprised of ODAR [26], [28] system. Each link modules can generate omni-directional force and torque with the non-aligned bi-directional rotors, and IMU, GPS modules are attached on the center of each link for the pose estimation of the system. The link modules of the system are connected each other using compliant cable, enabling to be acting as a 3DOF joint with wide range of motion. As the system has fully-actuated link modules, various control methods are available and one of the possible controller would be a decentralized impedance controller [3] as following

$$\begin{aligned} u_i &= R_i^T(m_i g e_3 + m_i \ddot{p}_{i,d} + k_d \dot{e}_{p,i} + k_p e_{p,i}) \\ \tau_i &= S(\omega_i) J_i \omega_i - k_R e_{R,i} - k_\omega e_{\omega,i} - k_I e_{I,i} \\ &\qquad - J_i(S(\omega_i) R_{i,d} \omega_{i,d} - R_{i,d} \dot{\omega}_{i,d}) \end{aligned} \tag{3.1}$$

where $\ddot{p}_{i,d}, \omega_{i,d}, \dot{\omega}_{i,d} \in \Re^3$ are desired acceleration, angular velocity and angular acceleration, $e_{p,i} := p_i - p_{i,d} \in \Re^3$,$\dot{e}_{p,i} := \dot{p}_i - \dot{p}_{i,d} \in \Re^3$ are position and velocity error, $e_{R,i} := (R_{i,d}^T R_i - R_i^T R_{i,d})^\vee \in \Re^3$, $e_{\omega,i} := \omega_i - R_i^T R_{i,d} \omega_{i,d} \in \Re^3$, $e_{I,i} := \int e_{\omega,i} + \gamma_I e_{R,i} dt \in \Re^3$ are rotation, angular velocity and integral error, $(\star)^\vee$ is the mapping from skew-symmetric matrix to a vector, and $k_d, k_p, k_R, k_\omega, k_I, \gamma_I \in \Re$ are control gains. Here, notice that the position estimation is taking an important role for the control, which also leads to the motivation of the estimation framework in this work. Also, the controller provides an error-tolerant property, since it is an impedance control having some extent of compliancy.

3.3 Pose and Posture Estimation

3.3.1 Estimation Algorithm via SCKF

For the state estimation of the f-LASDRA system, in this thesis, the SCKF algorithm in [14] is modified and extended to be used for multi dimensional constraint, and the constraint is applied using error-state kinematics to deal with quaternion states. The developed pose estimation algorithm is composed of two steps, a standard EKF and constraint application step, and both steps are described respectively in the following paragraphs.

Extended Kalman Filter

In the standard EKF step, the EKF state and covariance is updated with the measurement which can be expressed as

$$\hat{x}^E_{i,k}, P^E_{i,k} \leftarrow \text{EKF}(\hat{x}^E_{i,k-1}, P^E_{i,k-1}, y_{i,k}) \tag{3.2}$$

where we denote that the subscript $i \in \{1, ..., n\}$ is the link index, k is the step number of the filter, $\hat{x}_{i,k} \in \Re^{10}$ is the estimate of the state defined as

$$x_{i,k} := [p_{i,k}; v_{i,k}; q_{i,k}] \in \Re^{10}$$

where $p_{i,k}, v_{i,k}, q_{i,k} \in \Re^3$ are position, velocity and quaternion vector, and the superscript E means the variable resulting from the EKF. Also, $P_{i,k} \in \Re^{9\times9}$ is the covariance matrix of $\tilde{x}_{i,k}$, where $\tilde{x}_{i,k} := [\tilde{p}_{i,k}; \tilde{v}_{i,k}; \delta\theta_{i,k}] \in \Re^9$ is the error state for the EKF [39], [42], and $\tilde{p}_{i,k} := p_{i,k} - \hat{p}_{i,k}$, $\tilde{v}_{i,k} := v_{i,k} - \hat{v}_{i,k}$, and $\delta\theta_{i,k} \in \Re^3$ is the attitude error represented with angle vector which has relation of $q_{i,k} \otimes \hat{q}^{-1}_{i,k} \simeq [1; \frac{1}{2}\delta\theta_{i,k}]$, $\otimes$ is a quaternion multiplication operator, and $y_{i,k}$ is the measurement (e.g., GPS and IMU). Although there can be more state variables for the EKF state in (3.2) such as sensor biases, magnetic field vector, and so on, here we only consider position,

velocity, and quaternion vectors as those are the variables to be updated by the constraint applying process.

Constraint Application

In this step, the estimate result of EKF and the constraint information is used to obtain the estimation result coherent with the kinematic constraint given by the system configuration. The constraint application process can be described with pseudo-code as following where

Algorithm 1: Constraint application process

$j \leftarrow 0,\ \hat{\mathbf{x}}^C_{k,j} \leftarrow \hat{\mathbf{x}}^E_k,\ \mathbf{P}^C_{k,j} \leftarrow \mathbf{P}^E_k$
while $||\mathbf{c}_{k,j}(\tilde{\mathbf{x}}^C_{k,j})||_2 > \epsilon$ **do**
  $(\hat{\mathbf{x}}^C_{k,j+1}, \mathbf{P}^C_{k,j+1}) \leftarrow \mathrm{CA}(\hat{\mathbf{x}}^C_{k,j}, \mathbf{P}^C_{k,j}, \mathbf{c}_{k,j}(\hat{\mathbf{x}}^C_{k,j}))$
  $j \leftarrow j + 1$

$$\hat{\mathbf{x}}_k := [\hat{x}_{1,k}; \hat{x}_{2,k}; \cdots; \hat{x}_{n,k}] \in \Re^{10n}$$

is the stacked vector, n is the number of links of the system, $\mathbf{P}_k \in \Re^{9n \times 9n}$ is the stacked block diagonal matrix of covariance matrices $P_{i,k}$, $\mathbf{c}_k(\hat{\mathbf{x}}^E_k) \in \Re^{6(n-1)}$ is the constraint error with the state estimate $\hat{\mathbf{x}}^E_k$, CA() is the abbreviation of "Constraint Apply", the superscript C means the state and covariance resulting from the constraint application process, and the subscript j is the step index that is initialized with 0 and added 1 after every loop. For the aerial skeleton system, the constraint error $\mathbf{c}_k(\hat{\mathbf{x}}_k)$ is defined as

$$\mathbf{c}_k(\hat{\mathbf{x}}_k) = [c_{1,k}; c_{2,k}; \cdots; c_{n-1,k}] \tag{3.3}$$

where the element $c_{i,k}$ is meaning the difference of position and velocity estimate at the point of joint between i-th and $(i+1)$-th links, and can be expressed as

$$c_{i,k}(\hat{x}_{i,k}, \hat{x}_{i+1,k}) = \begin{bmatrix} \hat{p}_{i,k} + \hat{R}_{i,k} d_i \\ \hat{v}_{i,k} + \hat{R}_{i,k} S(\hat{\omega}_{i,k}) d_i \end{bmatrix} - \begin{bmatrix} \hat{p}_{i+1,k} - \hat{R}_{i+1,k} d_{i+1} \\ \hat{v}_{i+1,k} - \hat{R}_{i+1,k} S(\hat{\omega}_{i+1,k}) d_{i+1} \end{bmatrix}$$

where $i \in \{1, ..., n-1\}$, $\hat{p}_{i,k}, \hat{v}_{i,k}, \hat{\omega}_{i,k} \in \Re^3$ are position, velocity, and angular velocity estimate of i-th link at k-th step, $\hat{R}_{i,k} \in \mathrm{SO}(3)$ is the rotation matrix estimate, and $d_i \in \Re^3$ is the position vector from the i-th link center of mass to the right side joint. With an assumption of the symmetry of each link, the position vector to the left side joint can also be denoted as $-d_i$.

For the constraint application process, we also exploit the error state and error covariance as in usual EKF algorithms. The main advantage of using the error state in this work is the simple calculation of the constraint Jacobian which is described below the (3.4). By subtracting $\mathbf{c}_k(\hat{\mathbf{x}}_k)$ to the constraint error with nominal state $\mathbf{c}_k(\mathbf{x}_k)$, and with the small angle approximation of $\delta\theta_{i,k}$, the constraint equation can be expressed with the error state as following,

$$\tilde{\mathbf{c}}_k(\tilde{\mathbf{x}}_k) = [\tilde{c}_{1,k}; \tilde{c}_{2,k}; \cdots; \tilde{c}_{n-1,k}] \tag{3.4}$$

dimension of which is $\tilde{\mathbf{c}}_k(\tilde{\mathbf{x}}_k) \in \Re^{6(n-1)}$ and the each element $\tilde{c}_{i,k}$ is defined as

$$\tilde{c}_{i,k}(\tilde{x}_{i,k}, \tilde{x}_{i+1,k}) = \begin{bmatrix} \tilde{p}_{i,k} - S(\hat{R}_{i,k} d_i)\delta\theta_{i,k} \\ \tilde{v}_{i,k} - S(\hat{R}_{i,k} S(\hat{\omega}_{i,k}) d_i)\delta\theta_{i,k} \end{bmatrix} - \begin{bmatrix} \tilde{p}_{i+1,k} + S(\hat{R}_{i+1,k} d_{i+1})\delta\theta_{i+1,k} \\ \tilde{v}_{i+1,k} + S(\hat{R}_{i+1,k} S(\hat{\omega}_{i+1,k}) d_{i+1})\delta\theta_{i+1,k} \end{bmatrix}$$

where $i \in \{1, ..., n-1\}$, $\tilde{\mathbf{x}}_k := [\tilde{x}_{1,k}; \tilde{x}_{2,k}; \cdots; \tilde{x}_{n,k}] \in \Re^{9n}$ is the stacked error state

vector, and it can be expressed again as $\frac{\partial \tilde{\mathbf{c}}_k}{\partial \tilde{\mathbf{x}}_k} \cdot \tilde{\mathbf{x}}_k = 0$. Here, we define the constraint Jacobian $\tilde{\mathbf{C}}_k(\hat{\mathbf{x}}_k) := \frac{\partial \tilde{\mathbf{c}}_k}{\partial \tilde{\mathbf{x}}_k} \in \Re^{6(n-1)\times 9n}$ which can be easily obtained as the $\tilde{\mathbf{c}}_k$ has a form of analytic product with the error states.

We can now describe the function "ConstraintApply()" in the pseudo-code as following equations.

$$\begin{aligned}
\mathbf{P}^w_{k,j} &= \alpha e^{-j} \tilde{\mathbf{C}}_{k,j} \mathbf{P}^E_k \tilde{\mathbf{C}}^T_{k,j} \\
\mathbf{K}_{k,j} &= \mathbf{P}^C_{k,j} \tilde{\mathbf{C}}^T_{k,j} (\tilde{\mathbf{C}}_{k,j} \mathbf{P}^C_{k,j} \tilde{\mathbf{C}}^T_{k,j} + \mathbf{P}^w_{k,j})^{-1} \\
\tilde{\mathbf{x}}^C_{k,j+1} &= -\mathbf{K}_{k,j} \mathbf{c}_{k,j}(\hat{\mathbf{x}}^C_{k,j}) \\
\mathbf{P}^C_{k,j+1} &= (\mathbf{I} - \mathbf{K}_{k,j} \tilde{\mathbf{C}}_{k,j}) \mathbf{P}^C_{k,j} (\mathbf{I} - \mathbf{K}_{k,j} \tilde{\mathbf{C}}_{k,j})^T \\
&\quad + \mathbf{K}_{k,j} \mathbf{P}^w_{k,j} \mathbf{K}^T_{k,j}
\end{aligned}$$

where $\mathbf{P}^w_{k,j} \in \Re^{6(n-1)\times 6(n-1)}$ is the weakening covariance, an artificial noise for the constraint application that designed to decrease exponentially as the loop goes on, $\mathbf{K}_{k,j} \in \Re^{9n\times 6(n-1)}$ is the kalman gain, $\tilde{\mathbf{C}}_{k,j} := \tilde{\mathbf{C}}_k(\hat{\mathbf{x}}_{k,j})$, α is a constant parameter which is set to be 0.01 as in [14]. Then the error state is update by the product of kalman gain and the constraint error, and the covariance matrix is also updated to be used for the next loop. Then, the state $\hat{\mathbf{x}}_{k,j}$ is updated with the error state as

$$\begin{aligned}
\hat{p}^C_{i,k,j+1} &= \hat{p}^C_{i,k,j} + \tilde{p}^C_{i,k,j+1} \\
\hat{v}^C_{i,k,j+1} &= \hat{v}^C_{i,k,j} + \tilde{v}^C_{i,k,j+1} \\
\hat{q}^C_{i,k,j+1} &= \hat{q}^C_{i,k,j} \otimes \delta q_{i,k,j+1}
\end{aligned}$$

where $\delta q_{i,k,j+1} \simeq [1; \frac{1}{2}\delta\theta_{i,k,j+1}]$ is a quaternion error vector and normalized before the multiplication. The loop is continued while the constraint error satisfies $||\mathbf{c}_{k,j}(\tilde{\mathbf{x}}^C_{k,j})||_2 < \epsilon$ where ϵ is a small enough number. After the termination of the constraint application loop (suppose that $j = t$), the resultant state estimate $\hat{\mathbf{x}}^C_{k,t}$ can be exploited as the final pose estimation $\hat{\mathbf{x}}_k$ used for the control.

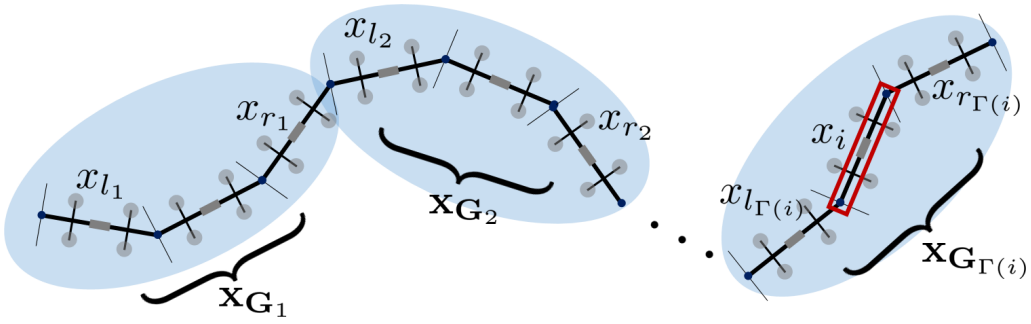

Figure 3-2: Illustrative figure of the notations for the semi-distributed version algorithm.

In this work, the result of constraint application is not fed back to the next step EKF state again, since with the original algorithm: 1) when the measurement data has large deviation from the constraint, measurement update and the constraint application process can conflict each other causing state updates with large magnitude and also can harm the stability of the estimation; 2) there exists an implementation issue that the EKF and constraint application for multiple links need to be synchronized at every time step, and it is not simply applicable for the system with distributed computing modules as the case of our system. Our estimation algorithm, then can be also interpreted as a projection of the EKF estimate result to the constraint using nonlinear constrained optimization problem similar to [43], where the constraint in this work is coming from the 3DOF joints of the system. The main difference of our proposed algorithm from the work in [43] is the concept of weakening covariance from SCKF, which enables systematically considering the linearization error by applying a vanishing artificial noise.

3.3.2 Semi-Distributed Version of Algorithm

In this work, a semi-distributed version of the algorithm in Sec. 3.3.1 is also presented to deal with the scalability issue when the number links is extremely large. In the semi-distributed algorithm, the entire links of the system are divided into several groups which contains some number of links where the number depends on

the computing performance of the on-board PC used for the system. Then, the constraint application process is performed for each group of links and the kinematic constraint can be obeyed at least among the links in a same group. To describe the proposed estimation scheme, let us define some notations (also refer to Fig. 3-2). First of all, each group of links are defined as a set and the γ-th group can be written as $\mathbf{G}_\gamma = \mathbf{G}_{\Gamma(i)} = \{l_{\Gamma(i)}, \cdots, i, i+1, \cdots, r_{\Gamma(i)}\}$ where γ is the index of the group increasing through the positive x direction in frame $\{L_i\}$ of each link, $\Gamma(i)$ is the index of the group that contains the i-th link, $l_{\Gamma(i)}, r_{\Gamma(i)}$ are the indices of the most left and the most right link of the γ-th group which contains i-th link, the number of elements in the set $\mathbf{G}_\gamma$ is denoted as $|\mathbf{G}_\gamma| := n_\gamma$, and the number of the groups in the entire system is denoted as s.

Then, the semi-distributed pose estimation algorithm, as a substitute for the constraint application process in Sec. 3.3.1, can be divided into local and global estimation steps, each will be explained in the paragraphs below.

Local Constraint Application

The local constraint application step shares the same function "CA()" in Algorithm 1, whereas the input for the function is changed as in Algorithm 2, where $\gamma =$

Algorithm 2: Local constraint application process

foreach $\gamma \in \{1, \cdots, s\}$ **do**
  $j \leftarrow 0,\ \hat{\mathbf{x}}^C_{\mathbf{G}_\gamma,k,j} \leftarrow \hat{\mathbf{x}}^E_{\mathbf{G}_\gamma,k},\ \mathbf{P}^C_{\mathbf{G}_\gamma,k,j} \leftarrow \mathbf{P}^E_{\mathbf{G}_\gamma,k}$
  while $||\mathbf{c}_{\mathbf{G}_\gamma,k,j}(\tilde{\mathbf{x}}^C_{\mathbf{G}_\gamma,k,j})||_2 > \epsilon$ **do**
    $(\hat{\mathbf{x}}^C_{\mathbf{G}_\gamma,k,j+1}, \mathbf{P}^C_{\mathbf{G}_\gamma,k,j+1}) \leftarrow$
      $\mathrm{CA}(\hat{\mathbf{x}}^C_{\mathbf{G}_\gamma,k,j}, \mathbf{P}^C_{\mathbf{G}_\gamma,k,j}, \mathbf{c}_{\mathbf{G}_\gamma,k,j}(\hat{\mathbf{x}}^C_{\mathbf{G}_\gamma,k,j}))$
    $j \leftarrow j+1$

$\Gamma(i) \in \{1, ..., s\}$, $\hat{\mathbf{x}}^E_{\mathbf{G}_\gamma,k} := [\hat{x}^E_{l_\gamma,k}; \cdots; \hat{x}^E_{r_\gamma,k}] \in \Re^{10n_\gamma}$ is the stacked vector of EKF pose estimates of links in γ-th group, $\mathbf{P}^E_{\mathbf{G}_\gamma,k} \in \Re^{9n_\gamma \times 9n_\gamma}$ is the covariance matrix of the error state $\tilde{\mathbf{x}}^E_{\mathbf{G}_\gamma,k} := [\tilde{x}^E_{l_\gamma,k}; \cdots; \tilde{x}^E_{r_\gamma,k}]$, similarly $\hat{\mathbf{x}}^C_{\mathbf{G}_\gamma,k} \in \Re^{10n_\gamma}$ and $\mathbf{P}^C_{\mathbf{G}_\gamma,k} \in \Re^{9n_\gamma \times 9n_\gamma}$ are the stacked state and covariance of pose estimates after constraint application,

$\mathbf{c}_{\mathbf{G}_\gamma,k}(\hat{\mathbf{x}}^E_{\mathbf{G}_\gamma,k}) \in \Re^{6(n_\gamma-1)}$ is the kinematic constraint error among the links in γ-th group, $\mathbf{c}^G_k(\hat{\mathbf{x}}^C_k) \in \Re^{6(s-1)}$ is the constraint error vector defined as

$$\mathbf{c}^G_k(\hat{\mathbf{x}}^C_k) := [c^G_{1,k}(\hat{x}_{r_1,k}, \hat{x}_{l_2,k}); \cdots ; c^G_{s-1,k}(\hat{x}_{r_{s-1},k}, \hat{x}_{l_s,k})]$$

which is required to be zero, and $c^G_{\gamma,k}(\hat{x}_{r_\gamma,k}, \hat{x}_{l_{\gamma+1},k})$ is the position and velocity difference between the right side tip of the most right link in the γ-th group and left side tip of the most left link in the $(\gamma+1)$-th group, similarly defined as $c_{i,k}(\hat{x}_{i,k}, \hat{x}_{i+1,k})$ in (3.3).

Global Constraint Application

After the local constraint application process in Algorithm 2, the kinematic constraint is satisfied within a group, but there still exist inconsistencies between the tips of the neighbouring group of links. To resolve these inconsistencies, the global constraint application process is executed which is expressed as following

$$\hat{\mathbf{x}}_k \leftarrow \mathrm{GCA}(\hat{\mathbf{x}}^C_k, \mathbf{P}^C_k, \mathbf{c}^G_k(\hat{\mathbf{x}}^C_k)) \tag{3.5}$$

where GCA() is the abbreviation of the "Global Constraint Application".

Since the computation amount of the global constraint application process is directly related with the number of link of the system, the highest priority of this process would be reducing the computation to achieve the system scalability. As the main cause of computation load in the constraint application process is the nonlinear kinematic constraint coming from the attitude vectors, in this process, we consider the attitude estimates $\hat{q}^C_{i,k}$ as given values and only update the position and velocity estimate. Then, the global constraint application process can be thought of as shifting the position and velocity vectors of each group to match the given kinematic constraint, that is, $\mathbf{c}^G_k(\hat{\mathbf{x}}^C_k) = 0$. Let us define the shifting vector $\Delta x_{\mathbf{G}_{\Gamma(i)}} := [\Delta p_{\mathbf{G}_{\Gamma(i)}}; \Delta v_{\mathbf{G}_{\Gamma(i)}}] \in \Re^6$ for each link, and the vector share same values

Table 3.1: Parameters used for the simulation

Descriptions	Values
Link length	1 [m]
Process noise	$\sim \mathcal{N}(0, \text{diag}[0.1I_3, 0.1I_3, 0.01I_3])$
Measurement noise	$\sim \mathcal{N}(0, \text{diag}[0.1I_3, 0.1I_3, 0.01I_4])$
Constraint apply stop condition	$\|\|\mathbf{c}_{k,j}(\tilde{\mathbf{x}}^C_{k,j})\|\|_2 < 0.01$ [m]

if the links are in the same group. To decide these shifting vectors, we solve for following constrained optimization

$$\begin{aligned} \min_{\Delta x_{\mathbf{G}_{\Gamma(i)},k}} \quad & \sum_{i=1}^{n} (\Delta x_{\mathbf{G}_{\Gamma(i)},k})^T \bar{P}^{-1}_{i,k} (\Delta x_{\mathbf{G}_{\Gamma(i)},k}) \\ \text{s.t.} \quad & \mathbf{c}^G_k(\hat{\mathbf{x}}_k) = 0 \end{aligned} \tag{3.6}$$

where $\bar{P}_{i,k} \in \Re^{6\times 6}$ is the covariance of $[p^C_{i,k}; v^C_{i,k}]$ which discarded quaternion vector from $x^C_{i,k}$, $\hat{\mathbf{x}}_k$ is the stacked vector of the final estimate element of which is calculated as $\hat{x}_{i,k} = \hat{x}^C_{i,k} + [\Delta x_{\mathbf{G}_{\Gamma(i)}}; 0] \in \Re^{10}$. The objective function means minimizing the Mahalanobis distance from the estimation $\hat{\mathbf{x}}^C_k$ to the $\hat{\mathbf{x}}_k$. Since (3.6) is the quadratic program with linear constraint, closed form solution can be easily obtained. Although this scheme is less optimal than the process in Sec. 3.3.1 as the attitude is not updated with the constraint apply, it does not need multiple loops of constraint application and can obtain updated estimate with a closed form solution.

3.4 Simulation

In this thesis, we perform simulation to verify the scalability issue of the devised estimation framework. In the simulation, 12 link aerial skeleton system is simulated with artificial measurements of IMU and GPS. The parameters used for the artificial measurements, EKF, and SCKF are summarized in Table 3.1. In addition to a white Gaussian measurement noise, we also applied bias with low frequency oscillation

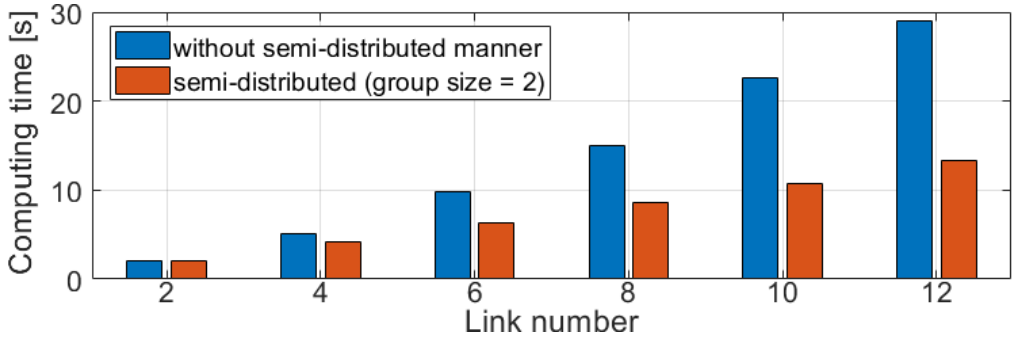

Figure 3-3: Comparison of computing time according to the link number increase and the usage of semi-distributed algorithm

(< 0.1Hz) for position and quaternion measurement with amplitude of 0.5m and 5° in axis angle, respectively, to emulate the GPS drift, biases of IMU.

Then, first of all, computing time increase with respect to the link number increase is checked with simulation, while comparing the semi-distributed version of the algorithm with non-applied algorithm. For both cases, simulation is conducted for 20 seconds, size of the group n_γ is set to be 2 for the semi-distributed algorithm, and the information of the whole links are insulted to the "CA()" function in Algorithm 1. The result is depicted in Fig. 3-3 and we can clearly see that the computation time of the semi-distributed algorithm grows much slower than the other, implying that the advantage of the algorithm further increases for the aerial skeleton with very large number of links.

For the second simulation, computation time and the accuracy of the estimation result are compared together, for the semi-distributed algorithm with different group sizes $n_\gamma \in \{2, 3, 4, 6, 12\}$. An illustrative figure of 12 link aerial skeleton during the algorithm is shown in Fig. 3-4 at simulation time $5, 10, 15$s, where each link module is drawn with a line segment. The figure shows the entire process of the proposed estimation framework from EKF to the final estimate after the global estimation process, and each case of $n_\gamma = 2, 12$ shows distinguishable difference of local estimation result (blue narrow dotted line), where two combined links are shown in the left column of figures. Then, the results of computation time and the

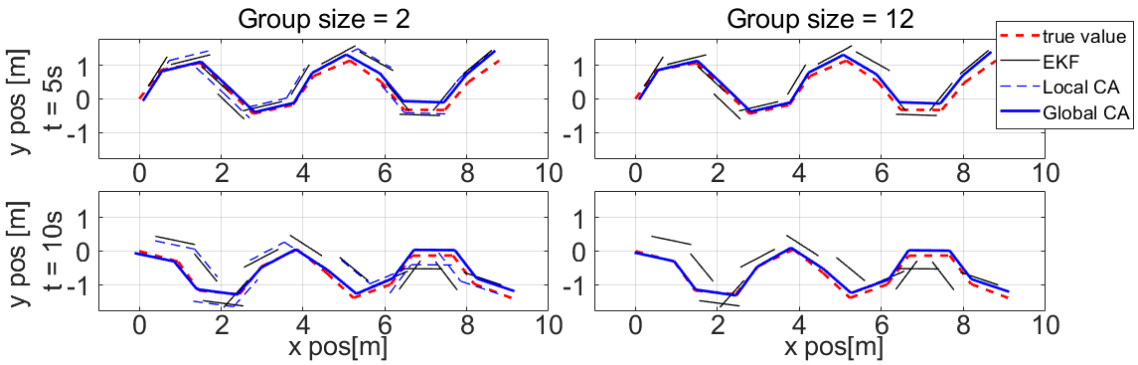

Figure 3-4: Illustrative figure of the process of the semi-distributed version of the constraint application algorithm.

Table 3.2: Computation time & estimation accuracy according to the group size increase

Group Size (n_γ)	Computing time [s]	Pos. error [m]	Quat. error
2	13.753	0.0377	0.0026
3	15.496	0.0370	0.0025
4	16.356	0.0369	0.0025
6	20.900	0.0360	0.0023
12	29.041	0.0322	0.0023

accuracy of estimation are presented in Table 3.2. Position and quaternion error in the table are meaning the two norm of the error (from the true value) average throughout the all links and all time steps in the simulation. The result shows that a slight increase of the estimation accuracy (about 15% and 12% for position and quaternion error respectively, from $n_\gamma = 2$ to $n_\gamma = 12$) is obtained with the increase of the group size, while sacrificing the computation time (more than twice from $n_\gamma = 2$ to $n_\gamma = 12$). This trade-off relation can be a reference for selecting the group size n_γ for the semi-distributed algorithm, considering the computing performance of the system and the desired extent of estimation accuracy.

3.5 Experiment

3.5.1 System Setup

To verify the proposed estimation framework, we implement the outdoor flight using the LASDRA system [3] containing three ODAR [26] system (1m length, 1.8kg weight for each) as link modules. The link modules are connected each other using a compliant PVC (polyvinyl chloride) cable which can provide high operation range. For the actuation of the system, DJI snail propulsion system is exploited with 6048-3D propeller that can generate bi-directional thrust upto about 8N. In the LASDRA system, both Raspberry Pi 3 and Pixhawk 2.4.8 are used as computing modules, one Raspberry Pi mounted on the middle link and three Pixhawks are attached on the center of every link modules. The Raspberry Pi takes a role of the main PC of the system, sending desired pose command, and collecting all the state and covariance information from the EKF running on each Pixhawks for the computation of SCKF algorithm. Due to the limited computing power of the onboard PC, the update rate is set to be 20Hz, yet, for the result of SCKF algorithm, only a difference from the EKF result is transferred to the Pixhawk so that the fast update rate of the EKF is not harmed while relatively slowly applying the constraint information. Then, on each Pixhawk, desired pose command and the updated state estimate with constraint application are received, EKF and controller are calculated with 500Hz and finally PWM signal is generated to run the rotors. For the power supply of motors, 4S LiPo battery attached on each link is exploited and the battery is also used to supply power for all the computing modules after stepdowning the voltage to 5V using the battery eliminator circuit (BEC). For the sensing modules of the system, IMU's inside the Pixhawk and GPS modules (U-blox NEO-M8) are used and those are mounted on every links of the system. Here, RTK (real-time kinematic) GPS can be another option for a sensing module which provides much more accurate position measurement than a general GPS, yet, as the main purpose of the system

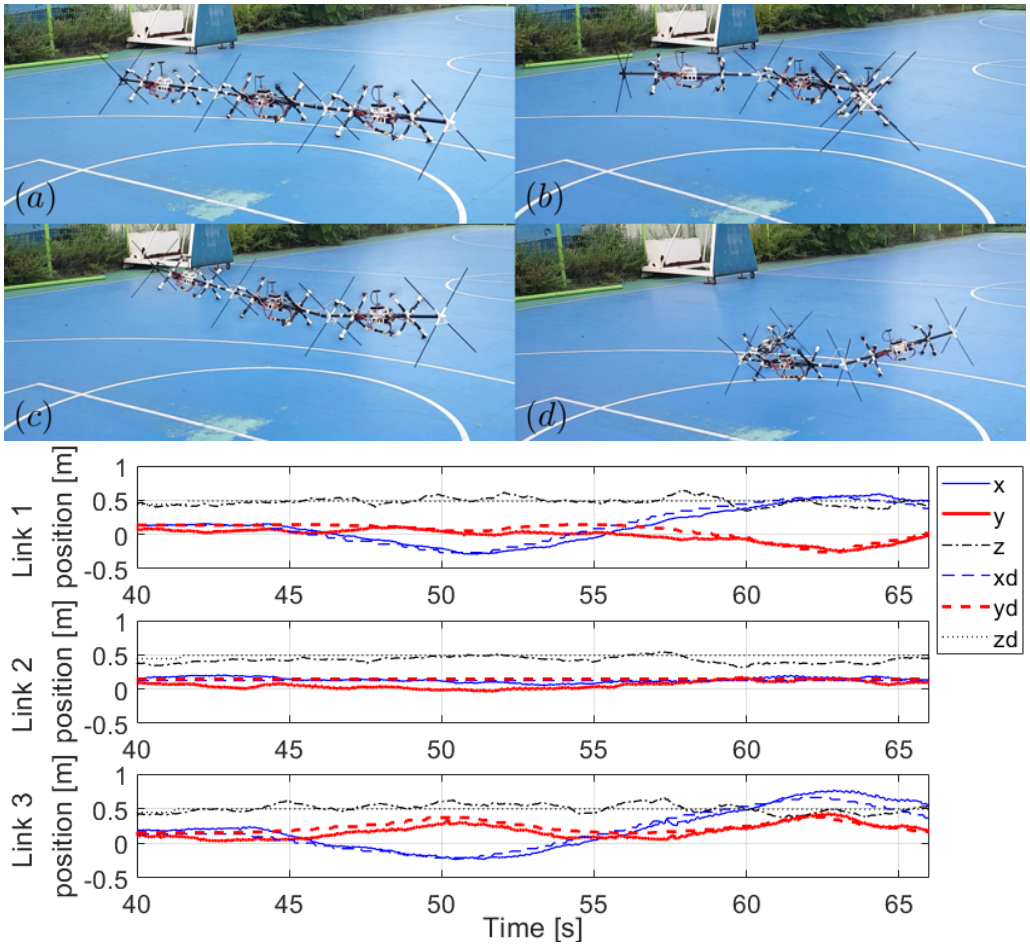

Figure 3-5: Snapshots and position tracking result of 3 link LASDRA outdoor flying experiment. Solid line: estimated position, dashed line: desired position.

prototype is verifying the performance of the proposed estimator, not achieving the best flight performance, we do not consider using of it. Exploiting the RTK-GPS and implementing the aerial skeleton with further accurate and agile motion would be one of our future work.

3.5.2 Experiment of SCKF-Based Estimation Algorithm

Then, with the constructed system above, outdoor flight experiment is performed as depicted in Fig. 3-5. After hovering for a few seconds, a pose trajectory is given to the system so that the system behaves like bending its configuration in forward

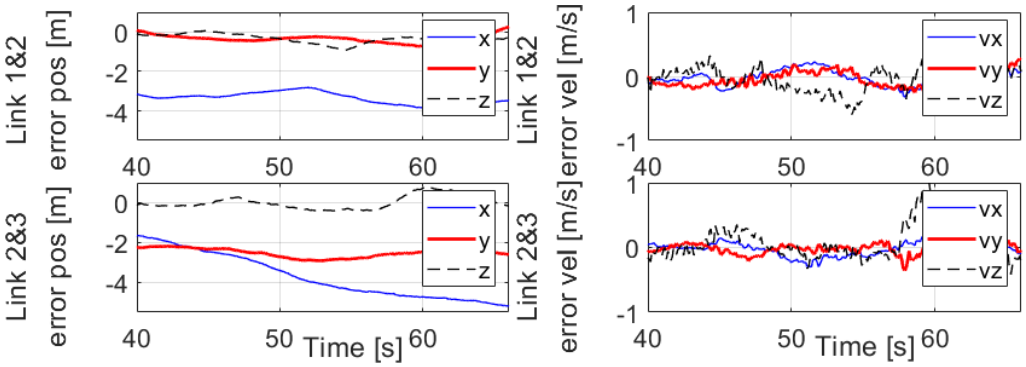

Figure 3-6: Constraint error (position and velocity difference between tips of neighbouring links) before the constraint application process.

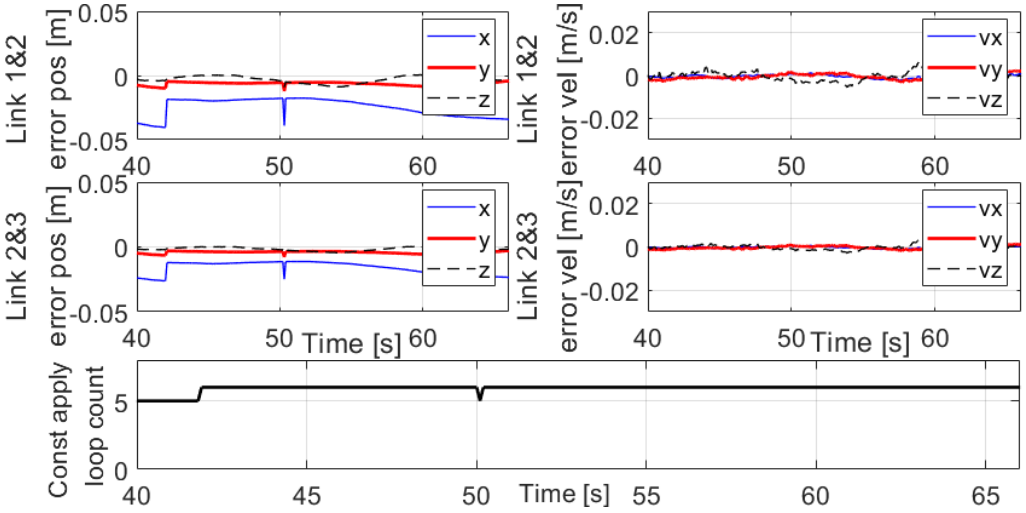

Figure 3-7: Constraint error (position and velocity difference between tips of neighbouring links) after the constraint application process, and the number of the constraint application loop run.

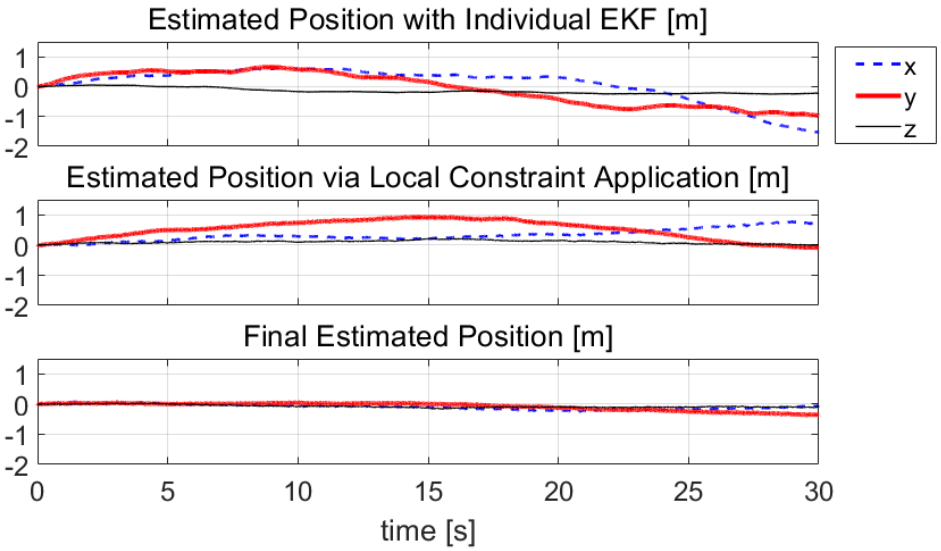

Figure 3-8: Position estimation result (4th link) while keeping the system static on the ground: (top) estimation result with individual EKF; (middle) estimation result via local constraint application; (bottom) final estimation result via semi-distributed algorithm.

and backward direction. The result of the flight is shown in the bottom of the Fig. 3-5. The RMS error between the desired and estimated position of each link is 11.61cm, 13.40cm, 12.57cm respectively. Although the position tracking error is not small, there was no issue at generating desired pose and posture, due to the error-tolerant property of the impedance control. Also in Fig. 3-6 and 3-7, the constraint errors before and after the SCKF algorithm are described, and these show that the constraint error is well regulated under 5cm and 2cm/s while there exist large constraint errors before the algorithm due to the inaccuracy of the GPS, and gyro noise. There are small jump and peak in the position constraint error near $42, 50$s and this happened due to the change of the loop number of constraint application process as shown in the bottom plot in Fig. 3-7. This can be alleviated if the loop stop condition $||\mathbf{c}_{k,j}(\tilde{\mathbf{x}}^C_{k,j})||_2 < \epsilon$ is set much tightly, where the ϵ in the experiment is set to be 0.05 to avoid an excessive computation.

We also tested the proposed estimation algorithm for a system with fixed position and identical attitudes for all links, so that the performance of the estimation can

be observed with the knowledge of true pose and posture. Here, we exploited seven link f-LASDRA system for the experiment, and $\{1, 2\}$, $\{3, 4, 5\}$, $\{6, 7\}$ links are comprising each local group for the semi-distributed algorithm. The experiment results are shown in Fig. 3-8, where the estimated position of the 4th link is shown as a representative, and the estimated position via individual EKF, local constraint application, and the final result through the semi-distributed version of the algorithm are depicted in order. The standard deviations of each estimation result of x, y, z position are turned out to be $[0.558; 0.542; 0.098]$ m, $[0.194; 0.311; 0.052]$ m, and $[0.081; 0.125; 0.046]$ m, respectively. We can see that as the number of link engaged for the estimation is increased, the estimated position of the link remained more closely to the initial position, which means a better estimation performance.

3.6 Conclusion

In this chapter, we present a novel pose and posture estimation framework of aerial skeleton system for outdoor flying only using IMU and GPS sensors. To enforce the kinematic coherency of the individual EKF estimates, we apply the kinematic constraints of the aerial skeleton system to the EKF estimates of all the links through SCKF-based constraint application process, thereby, enforcing the kinematic coherency of the skeleton system. Also, a semi-distributed version of the obtained estimation framework is presented to address the issue of scalability, so that required computation amount for the algorithm can be less affected by the increase of the link number. The proposed estimation framework is then verified with real outdoor flying experiments and simulation studies.

Chapter 4

CPG-Based Motion Generation

4.1 Introduction

Since one of the primary motivation of the flying LASDRA system is entertainment, it is important how its flying motion looks to the people, and our another goal would be generating motion that seems natural to spectators. In this chapter, we describe about the natural motion generation framework for the LASDRA system, and also expand and generalize it to apply for other robotic systems conducting complex environmental interaction. First, before describing about what is a natural motion generation, we divide the system motion into "shape motion" and "body motion" where the former is a motion seen from the system body frame, the latter is the motion of the system body frame seen from the inertial frame, and the sum of these motions can describe the entire motion of each link of the system. Then, we define a "natural motion" for the LASDRA system as a motion comprised of a shape motion that can be observed from similar counterpart of living creatures, and the body motion in accord with an expected physics with the given shape motion.

To generate shape motion of the system, we adapted **central pattern generator** (CPG), which is defined as a biological neural circuit that can produce coordinated motoneuron activities without sensory feedback [15]. CPG has been ex-

tensively used for controlling biomimetic robots including biped [44, 45], quadruped [46, 47], crawling [48, 49] and swimming robots [50], while being also used for other robots that requires rhythmic motion such as industrial robots [51], promoted by such a nice properties as robust limit cycle behaviour against perturbation, modulation of the motion using only a few parameters, smooth gait transition, and feasibility of distributed implementation, to list a few. These are also considered as good properties to be used as a tool for motion generation of LASDRA system, which has high degree of freedom and requires motion smoothness for the flight stability. Especially for the LASDRA system, we formulated a CPG model to mimic anguilliform motion (e.g., eel or lamprey-like swimming motion), since the shape of those species is most similar to that of the LASDRA system, and their main habitat is underwater which enables free 6DOF motion unlike the species on the ground. The information of anguilliform motion is obtained from biological experiments [52, 53], and the CPG model is formulated based on phase oscillator [54, 48]. In addition, to generate body motion endowed with naturalness, we simulated the system conducting anguilliform shape motion in the underwater environment with simple drag force model, and we obtain the body motion from the simulated pose and linear/angular velocity. Then, the combined motion of the shape and body motion produces final target pose for each link of the system, endowing it with a natural biomimetic motion.

For the CPG controlled robot, the CPG parameters and the resulting body motion from that parameter is connected via rather complicated dynamics and environmental interaction. Therefore, it is typically impossible to directly encode given body motion into CPG parameters. To resolve this issue, in this chapter, **inverse CPG model** is obtained so that it is possible to directly decide CPG parameters that generate desired resultant motion while incorporating complex environmental interaction. The inverse model, which takes desired motion as an input and provides CPG parameter set as an output, is obtained by data-driven approach with numerous simulation, and the model is constructed based on neural network. However,

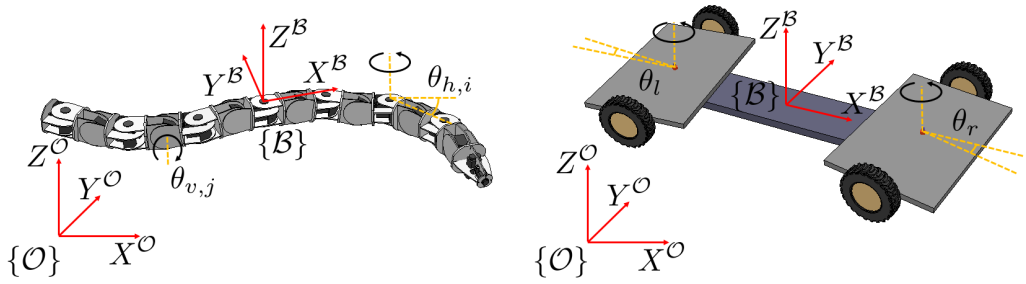

Figure 4-1: Two robotic systems, snake-like robot (left) and pivotboard (right) system, considered to develop and verify proposed CPG-based control framework. Also shown are the inertial and body coordinate frames, $\{\mathcal{O}\} := \{X^{\mathcal{O}}, Y^{\mathcal{O}}, Z^{\mathcal{O}}\}$, $\{\mathcal{B}\} := \{X^{\mathcal{B}}, Y^{\mathcal{B}}, Z^{\mathcal{B}}\}$; and horizontal, vertical, right and left joint angles, $\theta_{h,i}, \theta_{v,j}, \theta_r, \theta_l$.

obtaining this inverse model is not quite straightforward, as there exist dimension change between the desired motion and the CPG parameter causing information loss, and also, for some robots, neither body velocity nor acceleration of the robot is statically related with CPG parameters. Therefore, in this thesis, the inverse model is constructed along with stacked autoencoder to efficiently deal with the dimension decrease from input to output of the inverse model.

In this thesis, we also consider expanding and generalizing the proposed CPG-based motion generation framework including inverse CPG model to other robotic systems. To do so, we additionally consider two other robotic systems, snake-like robot [55] and pivotboard [56] (see Fig. 4-1). To explain the reason of selecting these systems, first, snake-like robot is operated using complex contact with the floor, and the pivotboard is operated with constraint force from the nonholonomic constraint. Both systems are operated by complex environmental interaction, which makes these systems not easy to apply direct motion planning or control design. The other point is that motion primitives to operated both systems are well known by biological inspection or human experience, and those are oscillating motion, being nice properties for CPG-based motion generation. Therefore, we believe that these systems would be nice examples to test our inverse CPG model based motion generation. Also, the fundamental relation of CPG parameters and the resultant

body motion is different for each system, i.e., parameters are almost statically related with body velocity of snake-like robot, but not for the pivotboard system, so the inverse model learning should be done differently. We will show that our concept of framework can be applied for various systems having fundamentally different relation with CPG parameter.

In case of using CPG-based motion generation framework for a robot in real world out of simulation, there might be model error or the change of the environment, and the generated motion can be different from expected. To overcome this issue, we propose a **CPG parameter adaptation law** based on backpropagation, so that the desired motion can be precisely generated even with the error of the obtained model or the environment change. The parameter adaptation using backpropagation is enabled by smooth neural network structure of the obtained inverse model. As a result, the combination of the inverse CPG model and the CPG parameter adaptation behaves as feedforward and feedback control for the system, enabling fast and robust generation of a desired motion.

There have been ample research of CPG modulation to modulate the resultant body motion, such as using sensory feedback [57, 58], parameter optimization [59, 60, 61], or other works surveyed in [62, 63], but only a few research considered modulating CPG parameters to produce a certain desired resultant motion (e.g., velocity or acceleration of the system center of mass), especially for a robot experiencing complex interaction with the environment. In [49], spine angles are controlled to fit to the desired trajectory using inverse kinematics, and made the robot to track the trajectory. However, the system capability is limited to following only desired curvature of the trajectory, and does not include direct modulation of CPG for forward speed modulation. Most recently in [45], optimal CPG parameter sets are found for some range of forward velocities, and enabled modulating the forward velocity using interpolated CPG parameters. However, as the parameters are optimized at a specific environment and did not consider adaptation of parameters, the desired speed generation is not robust against the change of the environment.

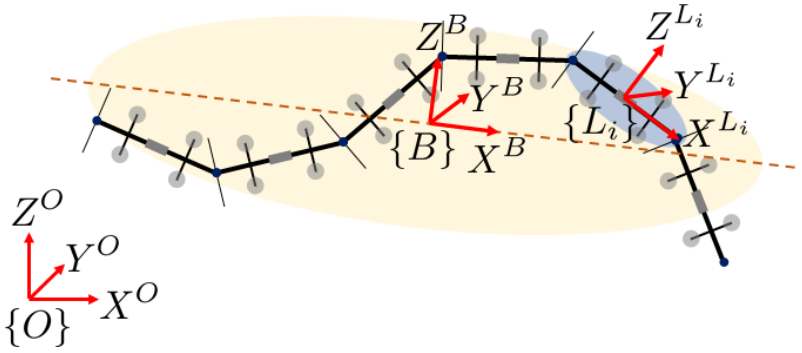

Figure 4-2: Inertial, body, and link coordinate frames defined for a description of CPG-based motion generation, $\{O\} := \{X^O, Y^O, Z^O\}$, $\{B\} := \{X^B, Y^B, Z^B\}$, $\{L_i\} := \{X^{L_i}, Y^{L_i}, Z^{L_i}\}$.

4.2 Description of Entire Framework

4.2.1 LASDRA System

The major application of the flying LASDRA system is the entertainment, being look natural and gratifying. This leads to the problem of how to make the system to generate a natural motion. Given this goal of motion generation, we set the motion naturalness as two elements, natural and biomimetic shape motion and accordance of the entire body motion to the expected physics.

Before the description, we define frames for the system as in Fig. 4-2. The link frame $\{L_i\}$ is the frame attached to i-th link, the body frame $\{B\}$ is a frame that represents the entire LASDRA system with its origin located at the center of mass of the system, and $\{O\}$ is the inertial frame. Then, we denote shape motion as the relative motion of LASDRA links in body frame $\{B\}$, and body motion as the motion of the body frame $\{B\}$ with respect to the inertial frame $\{O\}$. For the generation of natural and biomimetic shape motion, we adapt central pattern generator (CPG) which is frequently used in biomimetic robot control. The CPG is considered as a proper tool for the motion generation of f-LASDRA having high-DOF, since CPG can generate high DOF motion with few parameters, and also it provides robust limit cycle behavior, smooth motion modulation during the parameter transition. By

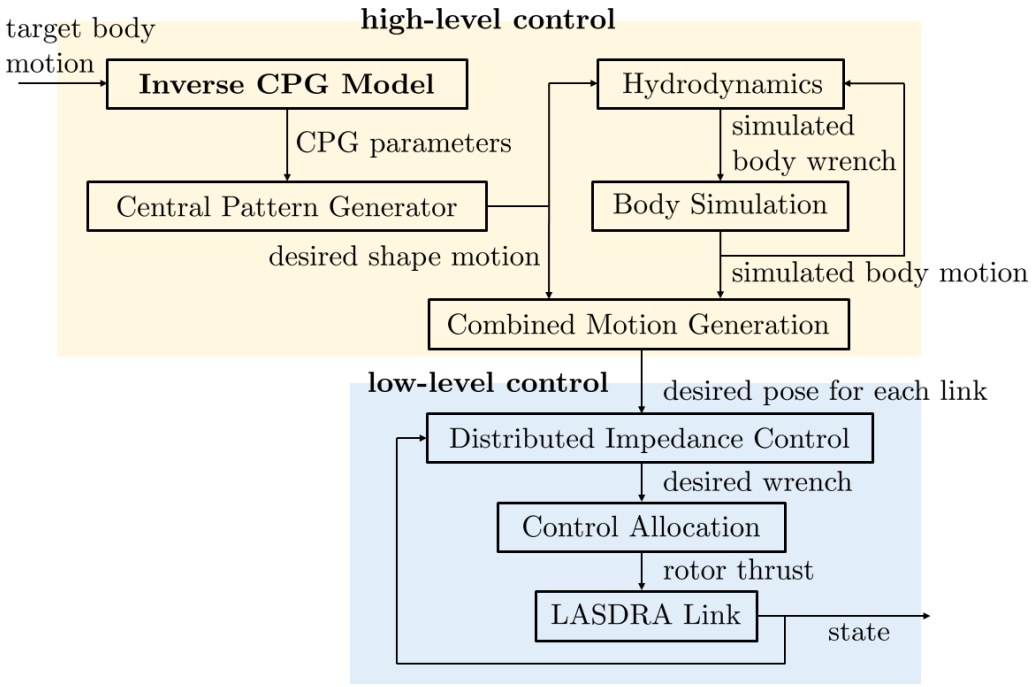

Figure 4-3: Block diagram of the control system of the flying LASDRA system including CPG-based motion generation.

using CPG, biomimetic motion is generated, in this work, eel-like undulating motion is selected regarding the serially linked system shape and its free flying property in 3D space. Then, the generated shape motion is simulated in some expected physics, which can be a water environment for our system, and the resultant system pose of the simulation becomes target trajectory pose. Although with the CPG model and the simulation, we do not know how to decide CPG parameters to generate desired velocity of the entire body. To deal with this issue, we learn the model from the generated body velocity to the CPG parameters using machine learning approach, and exploit it for generation of desired body velocity while conducting biomimetic undulatory motion. The entire framework of the motion generation for the system is depicted as block diagram in Fig. 4-3, and details will be described in following sections.

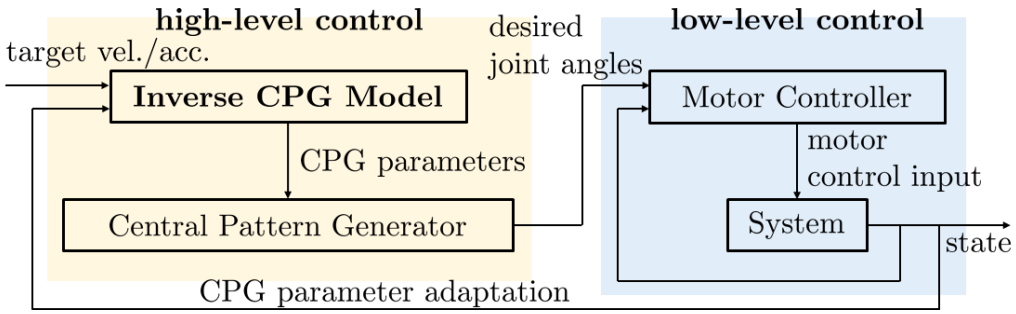

Figure 4-4: Block diagram of the entire control framework for snake-like robot and pivotboard based on CPG and inverse CPG model.

4.2.2 Snake-Like Robot & Pivotboard

Now let us describe mechanical construction of two robotic systems, snake-like robot [55] and pivotboard [56], that will be studied in this thesis for the generalization of the motion generation framework. The snake-like robot in this work consists of 15 link modules as in the left of Fig. 4-1. Each module is serially connected through revolute joints, axes of which are alternately directed toward horizontal and vertical direction. This design enables the system to mimic snake motion using well synchronized horizontal and vertical joint motion. The other system, pivotboard system (also known as snake board) is composed of one main deck and two foot plates, both foot plates are connected to tips of the main deck through revolute joints (see right of the Fig. 4-1). Two wheels are attached to each foot plate as usual skateboard, and by oscillating each foot plate with proper phase difference, the system can move using nonholonomic constraint.

The entire control framework is described as in Fig. 4-4, which has hierarchical structure composed of high-level and low-level controllers. Here, the main difference of the motion generation framework for general robotic systems (including the snake-like robot and pivotboard) from that of the LASDRA system is the existence of the simulation process. In contrast to the LASDRA system, both snake-like robot and pivotboard have real life counterpart to mimic (real snake and pivotboard driven by human), and motion naturalness can be achieved only by generating natural shape

motion. Therefore, the simulation process is no longer needed, and the expected natural body motion can be directly given by the interaction with real environment.

First of all, for the high-level controller, desired body motion command (e.g., body velocity or acceleration) is given as an input, and the inverse CPG block computes for the CPG parameters that can generate this desired body motion. Then, the system state or the information of the resultant body motion is fed back to the inverse CPG block again, and by doing so, CPG parameter adaptation is conducted enabling robust generation of the desired body motion. This adaptation is done using backpropagation law, which becomes possible due to our adoption of smooth MLP (multi layer perceptron) structure of the inverse CPG model. In the low-level controller, the motor control is done to track the desired joint angles computed from high-level controller using simple PD control. In this work, we are focusing on high-level controller, especially on the construction of the inverse CPG model, and CPG parameter adaptation law, which will be described in the following sections in detail.

This control framework has property of incorporating already known motion primitives that can operate the system, which could be given by living creatures, skills learnt by human, or by reinforcement learning. Therefore, this framework enables to achieve a control objective by exploiting motion primitives and feedback, although the system experiences complex environmental interaction and direct motion planning or control design to operate the system is not easy. Also, our proposed framework can be also applied nicely to the teleoperation, as the operator can provide command directly in desired motion which is much more intuitive than modulating CPG parameters leaving the process of learning the complex dynamics and environmental interaction for the operator.

4.3 CPG Model

In this section, CPG models designed for each robotic system are delineated. CPG models for all systems are formulated based on phase oscillator [54], and modified to control amplitude and bias of the oscillators as in [48]. We decided to choose CPG model of coupled oscillator type, which is the most abstract type among various models [62], as the main focus of this work is not on the study of neuron-wise behaviour and biology, and also the phase difference and topology made by connected oscillator units mainly affects the behaviour of the system. Details on CPG model of each robot will be explained in following subsections.

4.3.1 LASDRA System

Firstly, we describe the CPG model formulated for the natural biomimetic motion generation, and in this work we selected to mimic the eel-like undulating motion. CPG model is formulated based on phase oscillator [54] with several modifications for amplitude control [48] and 3D motion generation as following.

$$
\begin{aligned}
\ddot{a}_i &= k_a \left(\frac{k_a}{4}(\sigma_{a,i} A - a_i) - \dot{a}_i \right) \\
\ddot{b}_i &= k_b \left(\frac{k_b}{4}(\sigma_{b,i} B - b_i) - \dot{b}_i \right) \\
\dot{\phi}_i &= 2\pi f + \sum_j k_\phi \sin(\phi_j - \phi_i - \phi_{ij}) \\
w_i &= S(e_1) \begin{bmatrix} 0 \\ b_i + a_i \sin(\phi_i) \end{bmatrix} \\
R_{BL_i} &= e^{S(w_i)}
\end{aligned}
$$

where $A, B \in \Re^2$ are the input parameters of the model meaning target oscillating amplitude and bias; $\sigma_{a,i}, \sigma_{b,i}, \varphi_{ij} \in \Re$ are input shaping parameters that decides undulating shape given by experimental observation of eel swimming motion [52],

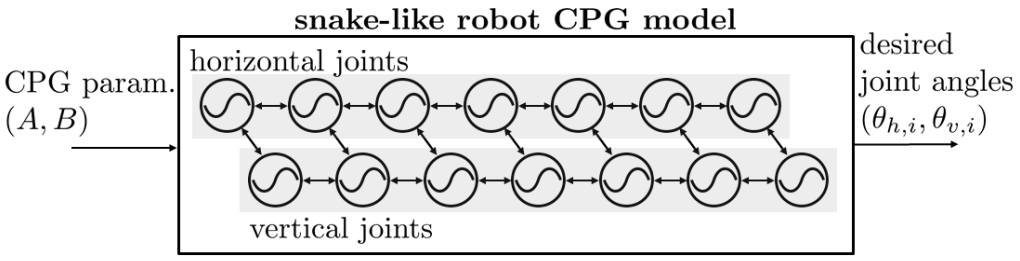

Figure 4-5: Diagram of CPG model for snake-like robot including oscillators and coupling topology depicted with wave patterns and arrows.

[53]; $a_i, b_i \in \Re^2$ are the intermediate parameters of amplitude and bias; $\phi_i \in \Re$ is the intermediate parameter meaning phase of the undulation; $f \in \Re$ is the given oscillating frequency; $k_a, k_b, k_\phi \in \Re$ are gain parameters; $w_i \in \Re^3$ is the axis angle to formulate the desired attitude; $R_{BL_i} \in \mathrm{SO}(3)$ is the desired attitude matrix being the final output of the proposed CPG model. In this work, f is not considered as an input parameter but set to be a constant value to avoid its redundancy with parameter A, since both parameters mainly affect for the forward velocity of the system. The formulated CPG model geometrically means rotation of i-th link through the axis of w_i and with the angle of $|w_i|$, resulting in horizontal undulation and bending with the first component of a_i, b_i and vertical undulation and bending with the second component of those states.

4.3.2 Snake-Like Robot

For the snake-like robot system, CPG model is formulated as following based on amplitude controlled phase oscillator [48] with modifications of adding a new state for bias modulation, and the topology change between oscillators for the synchro-

nization of horizontal and vertical joint angles,

$$
\begin{aligned}
\theta_{h,i} &= a_i \cos(\phi_{h,i}) + b_i \\
\theta_{v,i} &= -A_{v,i} \sin(\phi_{v,i}) \\
\ddot{a}_i &= k_a \left(\frac{k_a}{4}(\sigma_{a,i} A - a_i) - \dot{a}_i \right) \\
\ddot{b}_i &= k_b \left(\frac{k_b}{4}(\sigma_{b,i} B - b_i) - \dot{b}_i \right) \\
\dot{\phi}_{h,i} &= 2\pi f + \sum_{j \in \mathcal{N}_{h,i}} w_h \sin(\phi_{h,j} - \phi_{h,i} - \varphi_{h,ij}) \\
&\quad + w_{hv} \sin(\phi_{v,i} - 2\phi_{h,i} - \varphi_{hv}(A, B)) \\
\dot{\phi}_{v,i} &= 4\pi f + \sum_{j \in \mathcal{N}_{v,i}} w_v \sin(\phi_{v,j} - \phi_{v,i} - \varphi_{v,ij}) \\
&\quad - w_{hv} \sin(\phi_{v,i} - 2\phi_{h,i} - \varphi_{hv}(A, B))
\end{aligned}
$$

where $A, B \in \Re$ are the input parameters of the model meaning horizontal oscillating amplitude and bias; $\theta_{h,i}, \theta_{v,i}$ are the output parameters of the model meaning joint angles of horizontal and vertical joints; $i \in \{1, ..., 7\}$ is the index of horizontal or vertical joints; $a_i, b_i \in \Re$ are states meaning amplitude and bias; $\phi_{h,i}, \phi_{v,i} \in \Re$ are states meaning phase of oscillation for horizontal and vertical joints; $A_{v,i} \in \Re$ is a given parameter meaning amplitude for vertical direction; $\sigma_{a,i}, \sigma_{b,i} \in \Re$ are given input shaping parameters that decides undulating shape; $f \in \Re$ is the given oscillating frequency; $\varphi_{h,ij}, \varphi_{v,ij} \in \Re$ are the given targeting phase difference among horizontal joints and vertical joints that also decides undulating shape; $\varphi_{hv}(A, B) \in \Re$ is the targeting phase difference between horizontal and vertical joints which is a function of the input parameters A, B; $\mathcal{N}_{h,i}, \mathcal{N}_{v,i}$ are the sets of neighbouring joints of the i-th horizontal and vertical joints; and $k_a, k_b, w_h, w_v, w_{hv} \in \Re$ are gain parameters. In this work, f is not considered as an input parameter but set to be a constant value to avoid its redundancy with parameter A, since both parameters mainly affect for the forward velocity of the system.

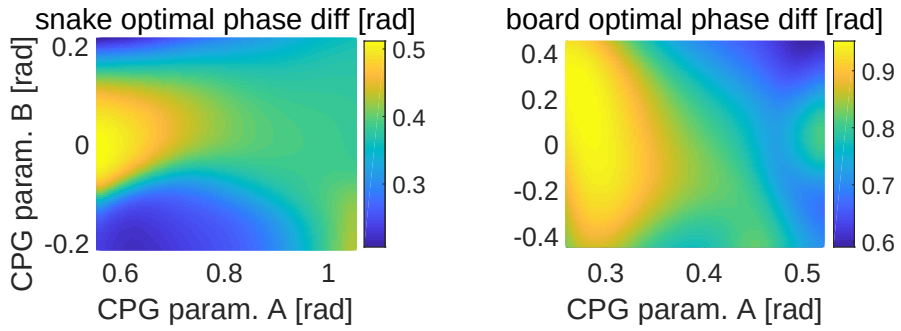

Figure 4-6: Optimal phase difference $\varphi_{hv}(A, B)$ and $\varphi(A, B, \dot{v}_d)$ $(\dot{v}_d \geq 0)$ providing maximum forward velocity for the snake-like robot (left) and the pivotboard (right).

The CPG model written above is designed to mimic serpentine motion of the snake, which conduct undulation mainly in horizontal direction and oscillate also in vertical direction to mimic 'sinus lifting' motion of snakes that controls the contact point of the body to maximize locomotion efficiency [55]. To do so, joint angles for horizontal and vertical joints are calculated separately with different frequencies, twice frequency for the vertical undulation. Couplings among horizontal joints or vertical joints are modelled with gain w_h, w_v, and also couplings between horizontal and vertical joints are modelled with gain w_{hv}, so that oscillation of all joints can be synchronized with desired phase differences. The topology of the designed CPG model is described in Fig. 4-5.

From the simulation, we found out that the parameter φ_{hv}, or the phase difference between horizontal and vertical joints behaves as a key parameter that critically affects the performance of locomotion, which implies that this value is required to be set carefully. To decide this parameter, we temporarily consider φ_{hv} as an input parameter just as A, B, and simulation is conducted with gridded parameter sets of A, B, φ_{hv}, slightly changing the value of each parameter. Then, the steady state body velocity is obtained for each parameter set, and from the simulated data, we get the function as below

$$\varphi_{hv}(A, B) := \arg\max_{\varphi_{hv}} v_x(A, B, \varphi_{hv}) \tag{4.1}$$

that selects $\varphi_{hv} \in \Re$ maximizing v_x, where $v_x \in \Re$ is the averaged steady state velocity in $X^{\mathcal{B}}$ (or forward) direction. By learning (A, B) to φ_{hv} model that maximizes the forward velocity from the simulation, we finally get an analytic function $\varphi_{hv}(A, B)$ that can be used directly. Here, we note that this function $\varphi_{hv}(A, B)$ is just part of the CPG model and not related with the inverse CPG model learning. The result of the obtained $\varphi_{hv}(A, B)$ function is depicted in the left of Fig. 4-6.

4.3.3 Pivotboard

Similar CPG model is formulated for the pivotboard system, also based on amplitude controlled phase oscillator [48] with additional state for bias modulation as following

$$
\begin{aligned}
\theta_i &= a_i \cos(\phi_i) + b_i \\
\ddot{a}_i &= k_a \left(\frac{k_a}{4}(A - a_i) - \dot{a}_i \right) \\
\ddot{b}_i &= k_b \left(\frac{k_b}{4}(\sigma_{b,i} B - b_i) - \dot{b}_i \right) \\
\dot{\phi}_r &= 2\pi f + w \sin(\phi_l - \phi_r - \varphi(A, B, \dot{v}_d)) \\
\dot{\phi}_l &= 2\pi f - w \sin(\phi_l - \phi_r - \varphi(A, B, \dot{v}_d))
\end{aligned}
$$

where $A, B \in \Re$ are the input parameters meaning target amplitude and bias; $\theta_i \in \Re$ is the output parameter meaning desired joint angle of the revolute joints; the index $i \in \{r, l\}$ denotes right and left foot plate; $\phi_i \in \Re$ is a state meaning oscillation phase; $\sigma_{b,r} := 1$ and $\sigma_{b,l} := 0$ are given input shaping parameters; $\varphi(A, B, \dot{v}_d) \in \Re$ is the targeting phase difference which is the function of A, B and $\dot{v}_d$; $\dot{v}_d \in \Re$ is the desired forward acceleration purpose of which will be explained below; $k_a, k_b, w \in \Re$ are gain parameters; and other parameters have same definition with those in the snake-like robot CPG model.

The CPG model above is designed based on oscillating motion primitives learnt and conducted by human. As the board can move forward by oscillating two foot

plates with slight phase difference, CPG model is formulated to mimic that motion. The target amplitude and bias of oscillation A, B are set to be input parameters for the model, and coupling between two revolute joint angles is modelled with the gain w, so that oscillation can be synchronized. The parameter $\varphi(A, B, \dot{v}_d)$, meaning phase difference between left and right joint, is decided by CPG parameters and also by the desired acceleration $\dot{v}_d$. When desired motion is to speed up, i.e., $\dot{v}_d \geq 0$, the parameter is selected to maximize the forward velocity of the system using the same approach with (4.1), and the result is show in the right of the Fig. 4-6. Meanwhile, it is found out from the simulations that when $\varphi = 0$, with any values of A, B, the velocity of the forgoing pivotboard converges to almost zero. Therefore, in case of deceleration or $\dot{v}_d < 0$, $\varphi(A, B, \dot{v}_d)$ is set to be zero, so that the system can quickly slow down, and the range of deceleration can be enlarged without bringing another input parameter for the CPG model. This kind of mode transition, or parameter modulation can be done smoothly thanks to the limit cycle property of the CPG.

4.4 Target Pose Calculation with Expected Physics

For the LASDRA system, the expected resulting body motion (eel-like swimming motion in water environment) from the CPG-generated shape motion is different from the motion resulting from real environment interaction (air environment), and no real life counterpart exists that can swim in the air (such as dragon). In this section, we describe the method of obtaining target pose through the simulation of the system in the expected physics, so that motion naturalness can be achieved by "acting" like interacting with an imaginary expected environment.

With the obtained shape motion given by the CPG model, the entire body motion is calculated by simulating the given shape motion in an expected physical environment, a water environment in this work. Assuming that inertial forces are dominant and fluid is stationary, simple water force model [64] is simulated, and

with the model, the force and torque applied to each link can be calculated as

$$\begin{aligned} f_i^{L_i} &= \frac{1}{2}\rho A_t C_{D,t} ||v_i^\top|| \cdot v_i^\top + \frac{1}{2}\rho A_n C_{D,n} ||v_i^\perp|| \cdot v_i^\perp \\ &= f_i^\top + f_i^\perp \end{aligned}$$

where $v_i^\top = I_1 v_{OL_i}^{L_i}, v_i^\perp = I_{23} v_{OL_i}^{L_i} \in \Re^3$ are the velocity elements of axial and perpendicular direction in the link frame; $I_1 := \text{diag}(1,0,0)$, and $I_{23} := \text{diag}(0,1,1)$. Then, force and torque applied to entire system can be obtained by summation as

$$\begin{aligned} f_{\text{cpg}}^B &= \sum_i R_{BL_i} f_i^{L_i} \\ \tau_{\text{cpg}}^B &= \sum_i (r_{BL_i}^B \times R_{BL_i} f_i^{L_i}) \end{aligned}$$

where $f_{\text{cpg}}^B, \tau_{\text{cpg}}^B \in \Re^3$ are force and torque applied to the entire system in body frame. Then the calculated force and torque is exploited for rigid body simulation as following,

$$\begin{aligned} m_{\text{exp}} \ddot{p}_{OB}^O &= R_{OB} f_{\text{cpg}}^B \\ J_{\text{exp}} \dot{\omega}_{OB}^B + S(\omega_{OB}^B) J_{\text{exp}} \omega_{OB}^B &= \tau_{\text{cpg}}^B \\ \dot{R}_{OB} &= R_{OB} S(\omega_{OB}^B) \end{aligned}$$

where $m_{\text{exp}} \in \Re$ and $J_{\text{exp}} \in \Re^{3\times3}$ are mass and inertia matrix of the expected dynamics of the system. Here, numerical integration for the dynamics simulation is done by partially (not including the external wrench) exploiting PMI (passive midpoint integrator) [65] to enforce the stability of the simulation. Then, the final target pose of each link can be obtained as below, by combining the shape motion

by CPG, and the simulated body motion

$$
\begin{aligned}
R_{OL_i} &= R_{OB} R_{BL_i} \\
\omega_{OL_i}^{L_i} &= R_{BL_i}^T \omega_{OB}^B + \omega_{BL_i}^{L_i} \\
&= R_{BL_i}^T \omega_{OB}^B + (R_{BL_i}^T \dot{R}_{BL_i})^\vee \\
p_{OL_i}^O &= \begin{cases} -p_{1,CoM} + p_{OB}^O & (i = 1) \\ \sum_{j=1}^{i-1} (R_j + R_{j+1}) l_h - p_{1,CoM} + p_{OB}^O & (i \geq 2) \end{cases} \\
v_{OL_i}^O &= \begin{cases} -v_{1,CoM} + \dot{p}_{OB}^O & (i = 1) \\ \sum_{j=1}^{i-1} (R_j S(\omega_j) + R_{j+1} S(\omega_{j+1})) l_h & \\ \qquad - v_{1,CoM} + \dot{p}_{OB}^O & (i \geq 2) \end{cases} \\
p_{1,CoM} &= \frac{1}{n} \sum_{i=2}^{n} \left(\sum_{j=1}^{i-1} (R_j + R_{j+1}) l_h \right) \\
v_{1,CoM} &= \frac{1}{n} \sum_{i=2}^{n} \left(\sum_{j=1}^{i-1} (R_j S(\omega_j) + R_{j+1} S(\omega_{j+1})) l_h \right)
\end{aligned}
$$

where $R_j = R_{OL_j}, \omega_j = \omega_{OL_j}^{L_j}$ for simplicity; l_h is the center to tip (positive x) vector in link frame; $n \geq 2$ is the number of links of the system. Here $\dot{R}_{BL_i}$ can be obtained analytically with Rodrigues' formula as in [66].

4.5 Inverse Model Learning

For each three rototic systems, inverse CPG model is learnt considering different relation properties of parameter to resultant body motion. Detailed explanation on inverse model learning is described in following subsections.

4.5.1 LASDRA System

In this subsection, the process of obtaining inverse CPG model is described which is required for the desired body velocity generation while conducting the biomimetic

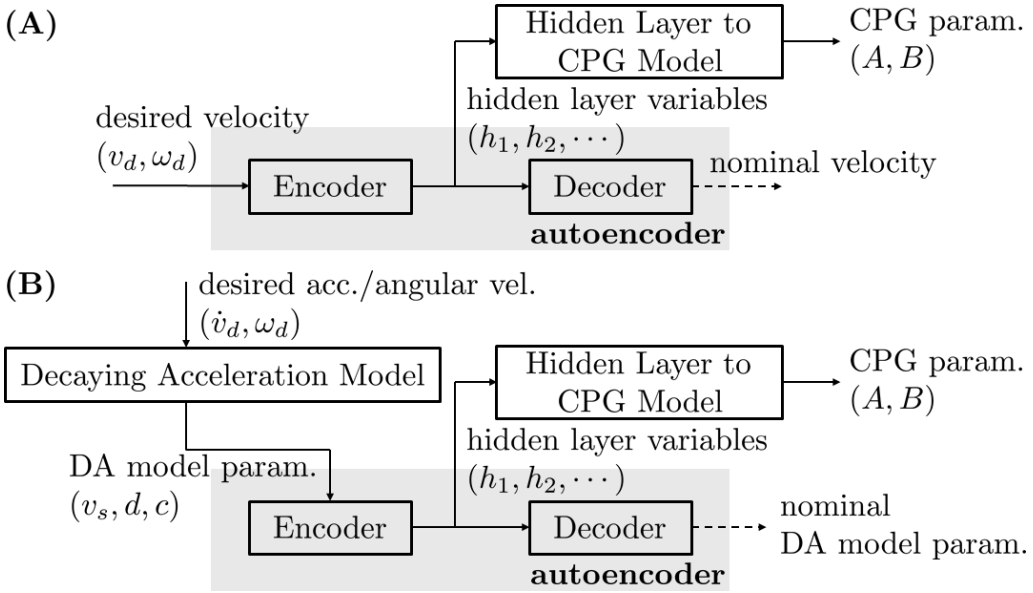

Figure 4-7: Diagram of inverse CPG model for (A) LASDRA and snake-like robot; and (B) pivotboard.

undulatory motion. From the simulation of the body motion in the expected physics environment, we found out that the velocity of the system in body frame converges to a certain value with a given set of CPG parameter. Considering this property, it is natural to obtain model between steady state body velocity and CPG parameters so that when learning the inverse model, we can get a CPG parameter set that generates certain desired body velocity. The diagram of the proposed inverse model is described in (A) of the Fig. 4-7. The target motion $\xi_d := [v_d; \omega_d; \theta_d] \in \Re^7$ is given as input for an autoencoder, where $v_d, \omega_d \in \Re^3$ are desired linear and angular velocity in frame $\{B\}$, and $\theta_d \in \Re$ is the desired undulating direction. Then, hidden layer variable with dimension of $\Re^4$ is obtained through the encoder which can be accounted as four extracted features from the input variable [67], and finally CPG parameters are provided through the 'hidden layer to CPG model'.

Here, let us explain about the reason and role of using autoencoder for the pro-

posed inverse model. For the simplicity of description, we define following variables

$$\eta := [A; B] \in \Re^4$$
$$\xi(\eta) := [v; \omega; \theta] \in \Re^7$$

where η is the vector of CPG parameters; and $\xi(\eta)$ is the averaged linear/angular velocity and undulating direction at steady state in $\{B\}$ coming out from the CPG generated motion with parameter η. Then, let us denote the inverse model that we want to get as $\Xi^{-1}(\xi(\eta)) \approx \eta$. However, as the dimension is decreasing with the inverse model $\Xi^{-1}(\cdot)$, the model will cause loss of information. Therefore, $\xi(\Xi^{-1}(\xi_d)) \not\approx \xi_d$ which means that even though we have CPG parameter from the inverse model with input ξ_d, the regenerated motion with the parameter does not always follows ξ_d. We believe that it is suitable to deal this kind of issue with the autoencoder, as it resembles the process of encoding and decoding of the autoencoder.

Then, in this work, autoencoder is learnt for the dataset $\xi(\eta)$ obtained by simulating the f-LASDRA system with uniformly sampled CPG parameters, with hidden layer in $\Re^4$. When we denote regenerated vector of ξ_d through encoding and decoding as $\mathrm{AE}(\xi_d)$, it can be interpreted as a projection of ξ_d to the feasible motion set, since the autoencoder is learnt with the real generated motion $\xi(\eta)$ from the robot, and $\mathrm{AE}(\xi_d)$ will be on the same manifold with the distribution of $\xi(\eta)$. The value of $\mathrm{AE}(\xi_d)$, which is explicitly obtained by autoencoder, can be usefully exploited for higher level controller, or also for the teleoperation noticing the operator of the gap of desired and feasible command.

After obtaining the autoencoder model and hidden layer variable using the encoder, a model of the hidden layer variable to CPG parameters is learnt using MLP with the hyperbolic tangent activation function, and then the whole inverse CPG model can be constructed. The result of the inverse model learning is described in the Fig. 4-8 showing input CPG parameter sets and the recovered parameter sets.

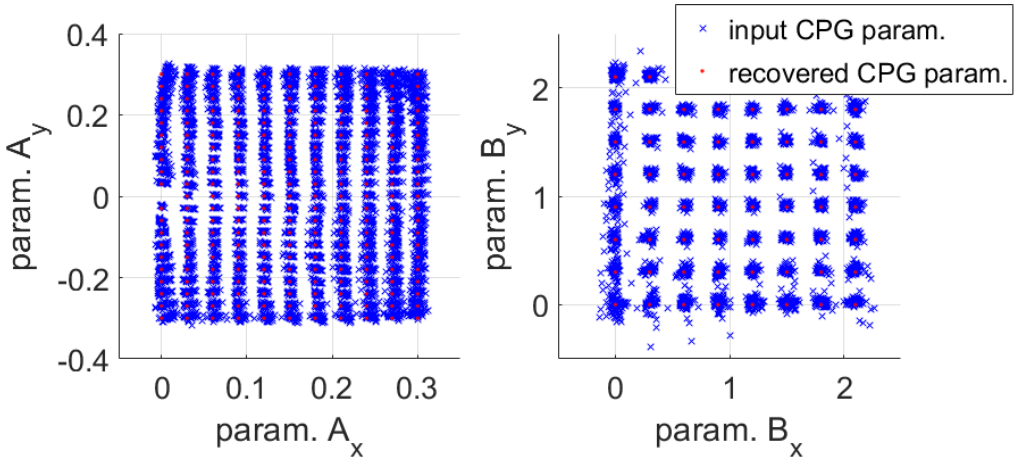

Figure 4-8: Comparison of input CPG parameter set and recovered CPG parameter set from the simulated velocity and the inverse model.

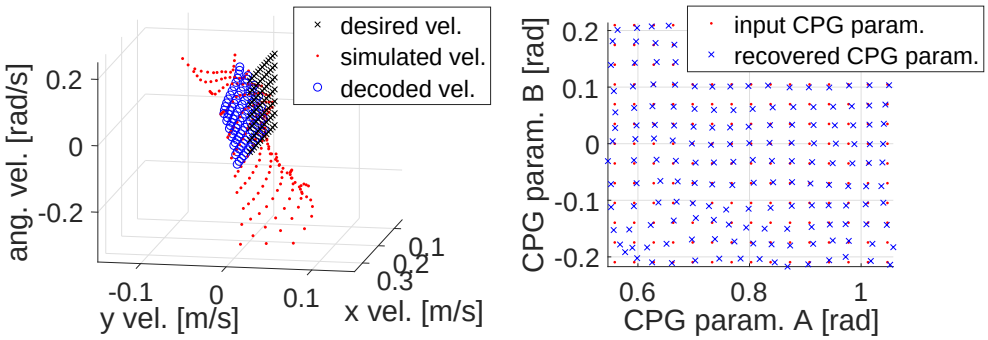

Figure 4-9: Autoencoder learning result of snake-like robot showing desired, simulated and decoded velocity (left); comparison of input CPG parameter set and recovered CPG parameter set from the simulated velocity and the inverse model.

Here, the recovered parameter sets are obtained by simulating with the input parameter sets, and deriving the CPG parameter sets that can regenerate the simulated motion using the inverse CPG model. Lastly in this work, CPG parameter sets with positive values of B_x, B_y are only used for learning thanks to the symmetry, so that required data and time for learning can be significantly reduced.

4.5.2 Snake-Like Robot

In case of snake-like robot simulation, velocity of the system in body frame converges to a certain value fast enough with a given set of CPG parameter. Considering this

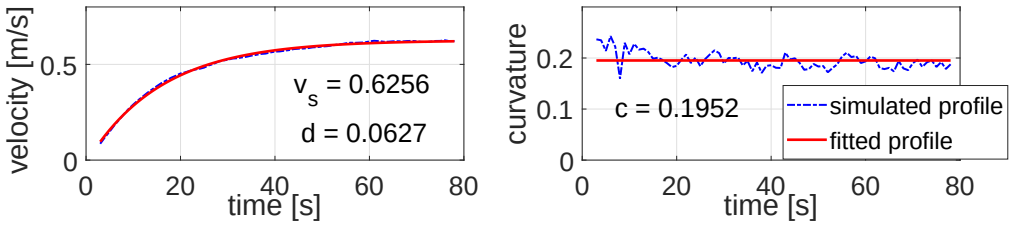

Figure 4-10: Velocity and curvature profile from the pivotboard simulation with $A = 20°, B = 10°$, and curve fitting result for DA model parameters.

property, it is natural to obtain model between steady state body velocity and CPG parameters so that when learning the inverse model, we can get a CPG parameter set that generates certain desired body velocity. The diagram of the proposed inverse model is described in (A) of Fig. 4-7. The desired linear and angular velocity $v_d \in \Re^2, \omega_d \in \Re$ are given as input for an autoencoder, hidden layer variables are obtained through the encoder, and finally CPG parameters are provided through the 'hidden layer to CPG model'.

The framework of obtaining inversion model is same with that of LASDRA system. Here, it is nice to describe and visualize the result of the inversion model learning with plots as in Fig. 4-9, since the dimension of the input, hidden and output layer of the stacked autoencoder is all less than or equal to three. The result of autoencoder model is depicted in the left of Fig. 4-9, showing that regenerated vector AE(ξ_d) from gridded ξ_d is projected on the plane of simulated velocity distribution, which means feasible velocity set. The result of the inverse model learning is described in the right of the Fig. 4-9 showing input CPG parameter sets and the recovered parameter sets obtained from inverse model and the simulated velocity with the input CPG parameter sets.

4.5.3 Pivotboard

Contrary to the LASDRA and the snake-like robot, from the pivotboard simulation, we observed that the body velocity of the system cannot be related statically with CPG parameters. To explain the simulated behaviour in more detail, with a

positive value of amplitude parameter A for the CPG model, the stationary pivot-board is first accelerated and the acceleration decays while forward velocity profile shows exponential convergence to certain value. Also, the bias parameter B for the CPG model is found out to be strongly and statically related with the curvature of the system position profile, and velocity in $Y^{\mathcal{B}}$ direction is almost negligible. The described behaviour is depicted as dotted blue line in Fig. 4-10.

Since we need to find parameters that can have static relation with CPG parameters, we choose to use following model from the simulation, and we call it 'decaying acceleration model' or simply a DA model,

$$\begin{aligned} \dot{v}(t) &= -d(v(t) - v_s) \\ v(t) &= (v(t_0) - v_s)e^{-d(t-t_0)} + v_s \\ \omega(t) &= cv(t) \end{aligned} \tag{4.2}$$

where $v_s, d, c \in \Re$ are DA model parameters meaning steady-state converging velocity, exponential decay rate, and curvature; $v(t), \omega(t) \in \Re$ are the averaged forward velocity and angular velocity during time $[t - \Delta t, t]$ in frame $\{\mathcal{B}\}$; Δt is the time period of the CPG oscillation; and t_0 is the time at the beginning of simulation with given CPG parameter set. Here, we use the averaged value during one period of the oscillation for $v(t), \omega(t)$ to get rid of the fluctuation of the value caused by the oscillation. Considering this model, simulations are conducted with both acceleration and deceleration cases, also with gridded CPG parameter sets slightly changing each values. Then, DA model parameters are obtained for each simulation using least square fitting for the velocity profile. A sample result of least square fitting with the DA model is shown as red line in Fig. 4-10.

What we ultimately want for the inverse model is the desired motion to CPG parameters relation, and the model can be divided into two steps: 1) desired motion to DA model parameters; and 2) DA model parameters to CPG parameters (see bottom of Fig. 4-7). Since the DA model parameters are known to have static

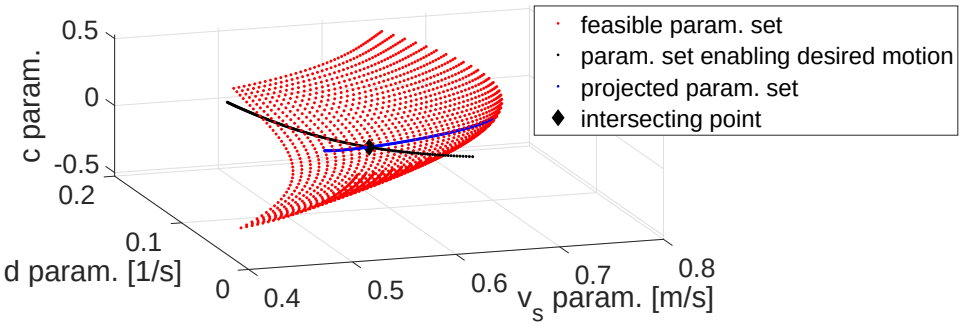

Figure 4-11: Result of finding DA model parameter set, where red dots are the feasible parameter set, black and blue line are the parameters set enabling desired motion and its projection on the feasible set.

relation with CPG parameters and its dimension is three, model for the second step can be obtained with the same method as in Sec. 4.5.2.

Then the remaining problem is the model for the first step, i.e., the model of desired motion to DA model parameters. Here, the desired motion is defined by $\dot{v}_d, \omega_d \in \Re$, the desired acceleration and angular velocity. First of all, the parameter c can be easily obtained from v, ω_d ratio referring to (4.2). Also, from (4.2), parameters v_s and d have relation of $v_s = v + \frac{\dot{v}}{d}$. Then, v_s, d relation with a given $v, \dot{v}_d$ and already obtained parameter c form a curved line (hyperbola) on a 3D space. From the autoencoder learnt with the parameter set of v_s, d, c, we know the feasible plane of the parameter set in 3D, so there would be a crossing point of the plane and the line (see Fig. 4-11). This point is the DA model parameter set that satisfies desired motion and feasibility of the parameter set itself, and can be simply obtained by common line search methods. The result of finding DA model parameter set with line search method is shown in Fig. 4-11. The red dots in the figure are the output of the autoencoder learnt with the DA model parameters from simulation, and the distributed plane means the feasible set of DA model parameters. Then, we finally obtain the whole inverse CPG model from the desired motion to the CPG parameters.

4.6 CPG Parameter Adaptation

Although we have inverse CPG model, in case of direct generation of the CPG-generated shape motion in real environment, it is inevitable to have model error for the learnt models or change of the environment, making it unable to precisely generate the desired motion with CPG parameters and the inverse model. Therefore, it is desirable to apply feedback or adaptation law for the parameters to generate desired velocity precisely. Since we have inverse CPG model constructed with MLP, the gradient of the parameter which is denoted as $\frac{\partial \eta}{\partial h}$ in (4.3) below can be analytically known. Also, as we use smooth hyperbolic tangent as an activation function of the MLP, the gradient $\frac{\partial \eta}{\partial h}$ is always smooth, and it helps the adaptation to be done smoothly by updating parameters using backpropagation. The equation for the CPG parameter adaptation can be written as following

$$\begin{aligned} \Delta\eta &= -\frac{\partial \eta}{\partial h}\left(k_p(h-h_d) + k_i \int (h-h_d)d\tau\right) \\ \eta &= \eta_{\text{raw}} + \Delta\eta \end{aligned} \tag{4.3}$$

where $h, h_d \in \Re^2$ are the hidden layer variables that encoded the current and desired DA model parameters; $k_p, k_i \in \Re$ are proportional and integral gains; $\Delta\eta \in \Re^2$ is the change of parameter set for the PI-like (proportional-integral) adaptation; $\eta_{\text{raw}} \in \Re^2$ is the raw parameter set given by the inverse model; and $\eta \in \Re^2$ is the final output CPG parameter set. In case of the pivotboard system, where the autoencoder input is the DA model parameter set, model parameter regarding current state of the system need to be obtained. To do so, the line search method in Sec. 4.5.3 is used again that finds DA model parameter satisfying constraints given by current state $v, \dot{v}$ and its own feasibility. Then the rest of the adaptation process can be done same with (4.3).

There is another option for the adaptation law of the CPG parameters, directly using the error from the desired motion (velocity or acceleration) for the parameter

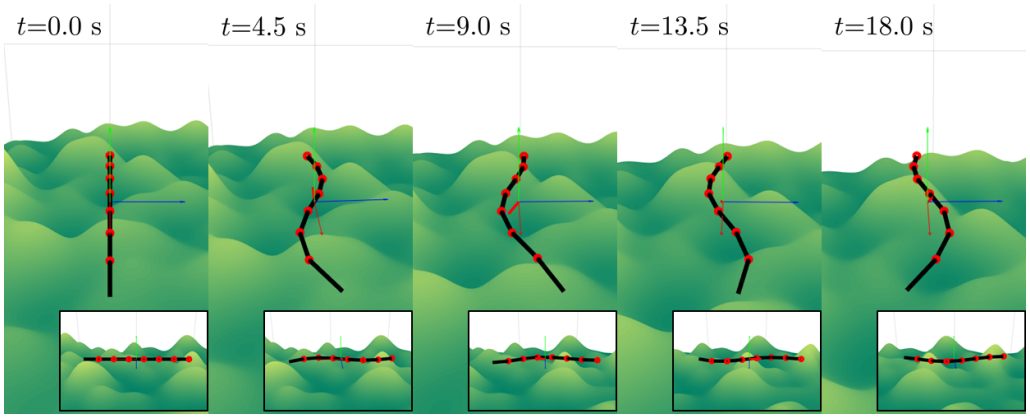

Figure 4-12: Snapshots of the simulation of CPG-based motion generation for 7-link f-LASDRA system.

update. Although this method is much more intuitive compared to (4.3), the error should be projected from $\Re^3$ to $\Re^2$ and there would be null component in the error. The integrated error in the null space would not affect the parameter update, but when the null space is changed with state change, the hidden accumulated error can pop out causing undesired motion. Therefore, we conduct parameter adaptation using the hidden layer variable which has the same dimension with the CPG parameter set, so that the risk stated above can be prevented.

4.7 Simulation

4.7.1 LASDRA System

To verify the newly devised CPG-based motion generation framework, simulation is conducted using 5-link and 7-link f-LASDRA system. A snapshot of the simulation of an horizontal undulatory motion is depicted as an example in Fig. 4-12. First of all, modulation of CPG parameters and the resulting target pose of each link is observe to verify the formulated CPG model in Sec. 4.3. From the plots in Fig. 4-13, we can check that with the modulation of CPG parameter A_x and B_x, amplitude or offset of the oscillating desired yaw angle of each link is smoothly changed. Also,

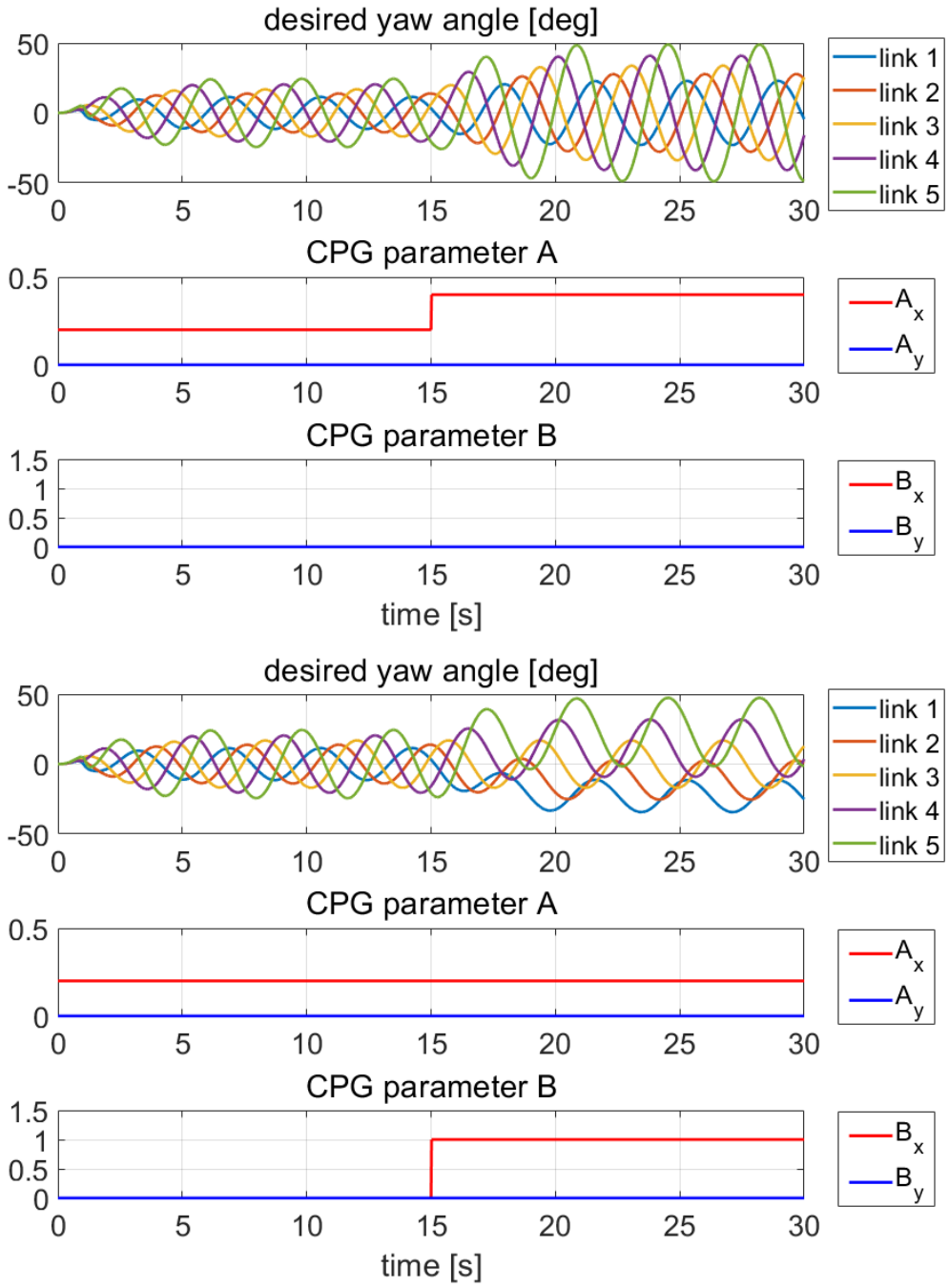

Figure 4-13: Simulation results of CPG-based motion generation: desired yaw angle for each link with (top) parameter A_x modulation; and (bottom) parameter B_x modulation.

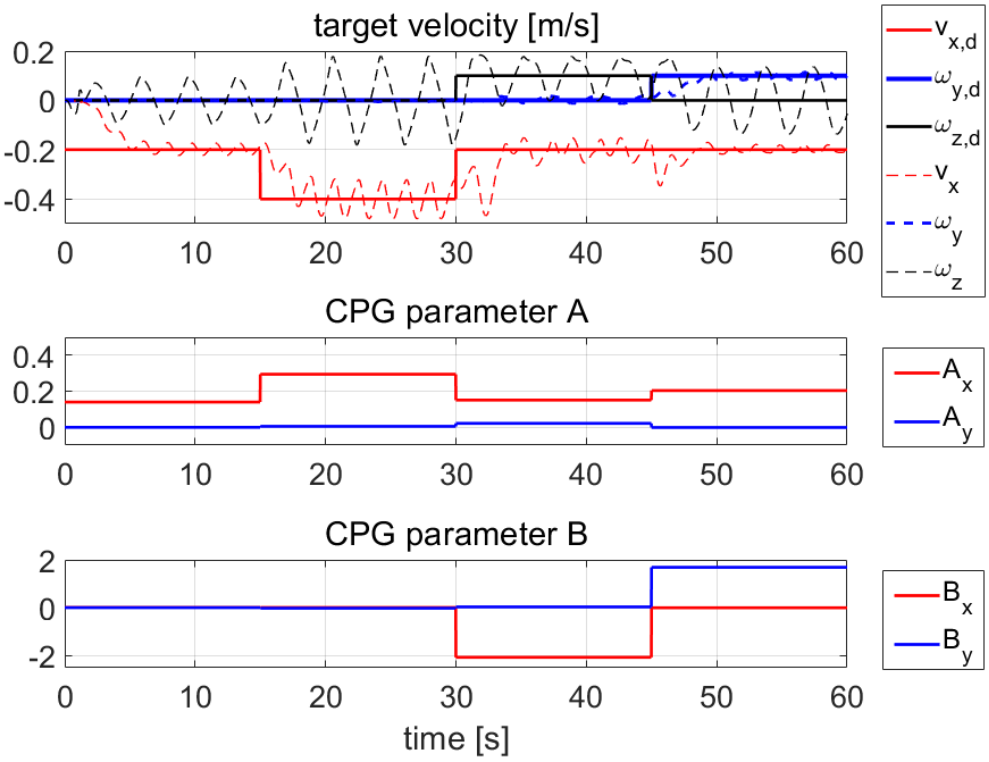

Figure 4-14: Simulation results of target motion generation with CPG-based motion generation framework including CPG inverse model

to verify the inverse CPG model learnt with MLP and autuencoder, simulation of generating target motion is conducted where the result is described in Fig. 4-14. We can see from the plot that CPG parameters are calculated by the inversion model, and the simulated body velocity follows the target motion in average, where the oscillating data plot is due to the undulatory motion from CPG.

4.7.2 Snake-Like Robot & Pivotboard

To check if the proposed CPG-based control framework can be properly extended to other general robotic systems, simulations with snake-like robot and pivotboard are also conducted using the open-source robot simulator V-REP with Vortex physics engine. Also, Matlab is exploited along with the simulator communicating through the remote API (application programming interface), for the control computation and data logging. The CPG model of each robot is simulated in the V-REP through the embedded script, and numerical update is done by PMI (passive midpoint integrator) [65] to enforce the stability of the simulation. Then in the Matlab script, state information of the CPG model is received, CPG parameter that achieves desired motion is computed with the inverse model and the adaptation law, and finally the parameter is sent back to V-REP embedded script. V-REP simulation runs with the rate of 100Hz, and the CPG parameter update is done at the same rate as the oscillation (1Hz for both systems).

First of all, to check the performance of our proposed framework, simulations of desired linear and angular velocity generation with CPG controlled snake-like robot are conducted. For the simulation of desired linear velocity generation, y (lateral) position is regulated to be zero, and the simulation result is shown in top of the Fig. 4-15, where the RMS (root mean square) error of the generated linear velocity and y position are 0.030 m/s and 0.140 m. Also for the simulation of desired angular velocity generation, linear velocity is maintained to be constant, and the simulation result is shown in bottom of the Fig. 4-15, where the RMS error of

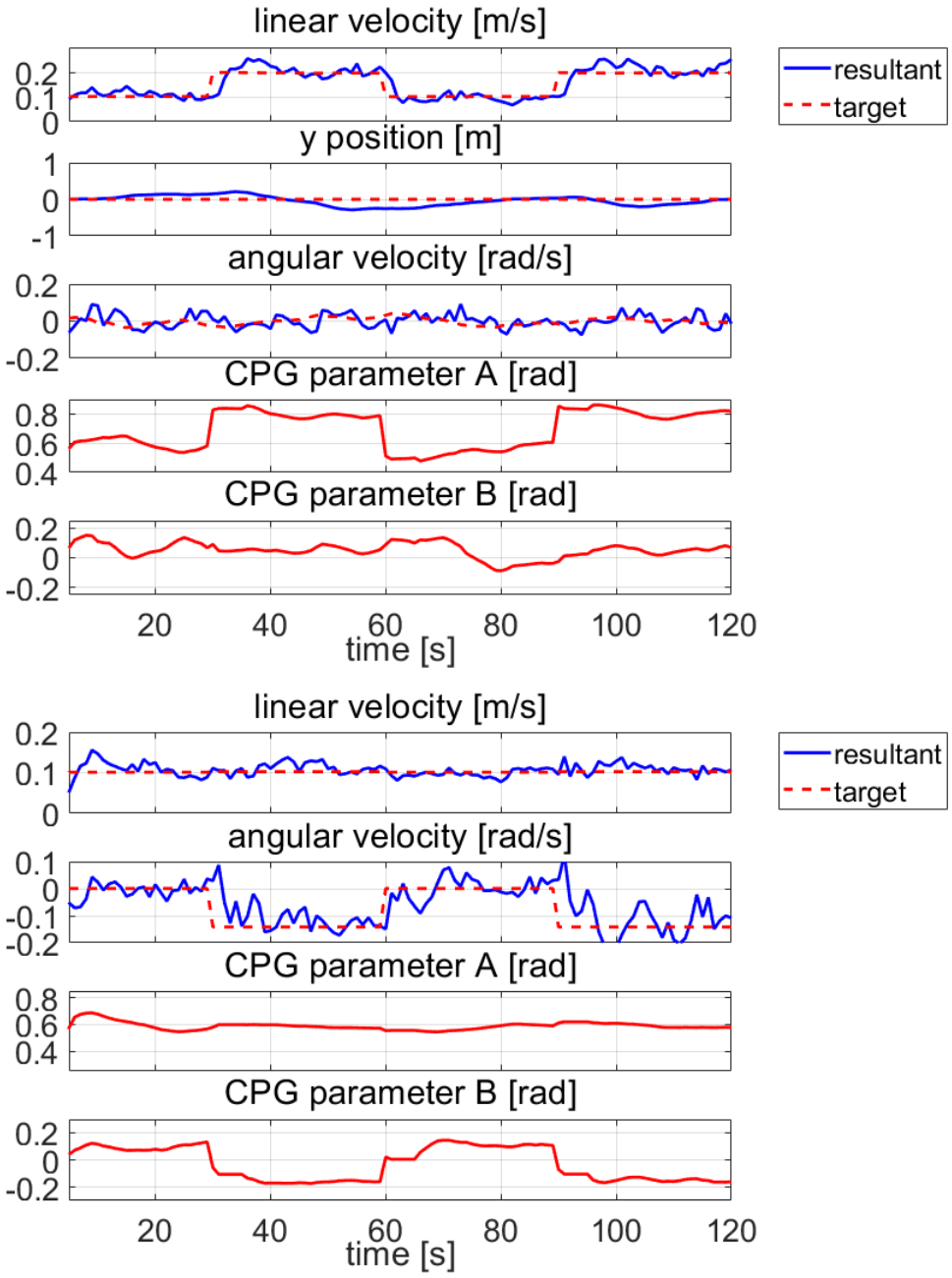

Figure 4-15: Simulation results of snake-like robot with various linear velocity (top) and angular velocity (bottom) command.

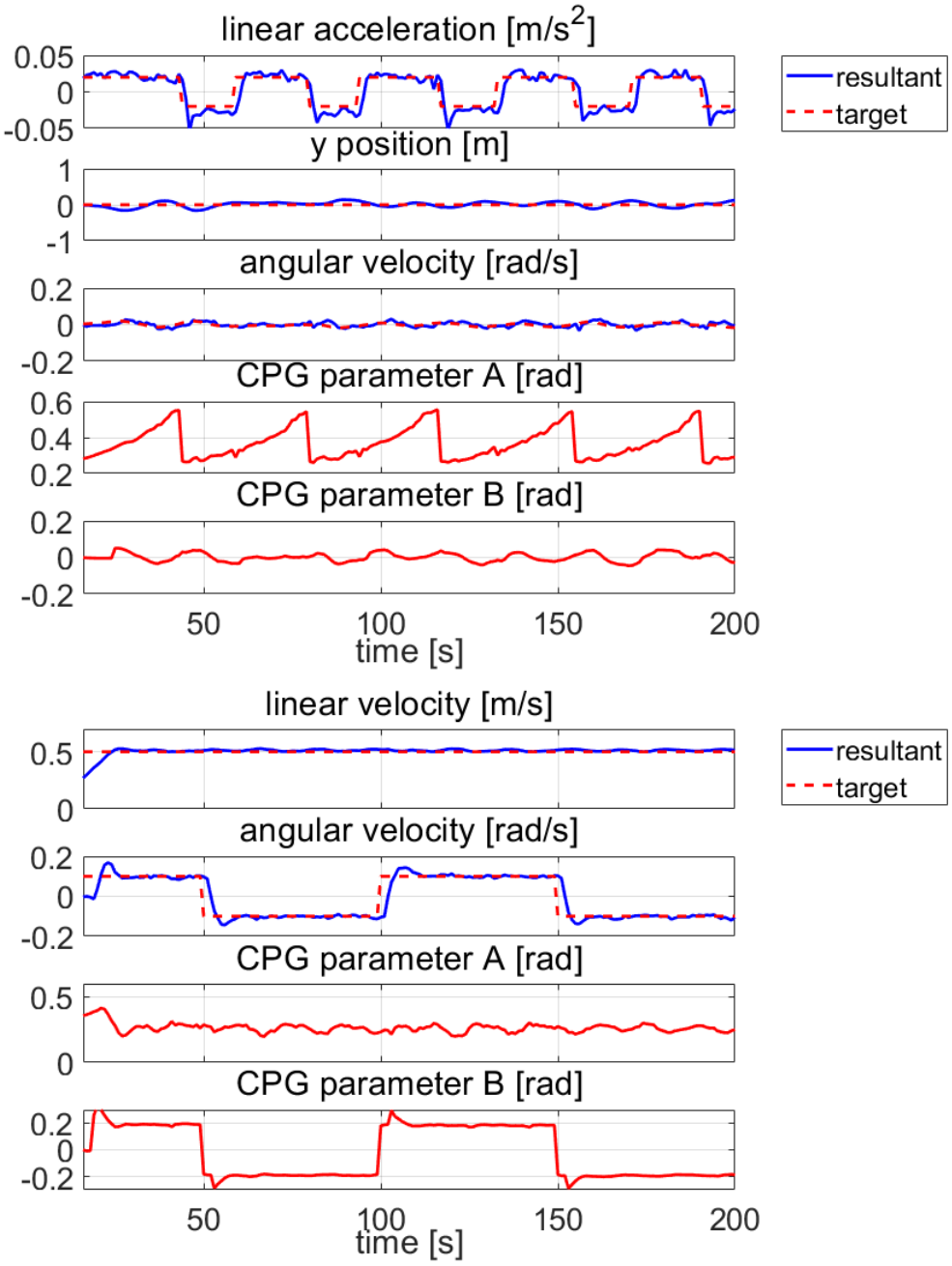

Figure 4-16: Simulation results of pivotboard with various linear acceleration (top) and angular velocity (bottom) command.

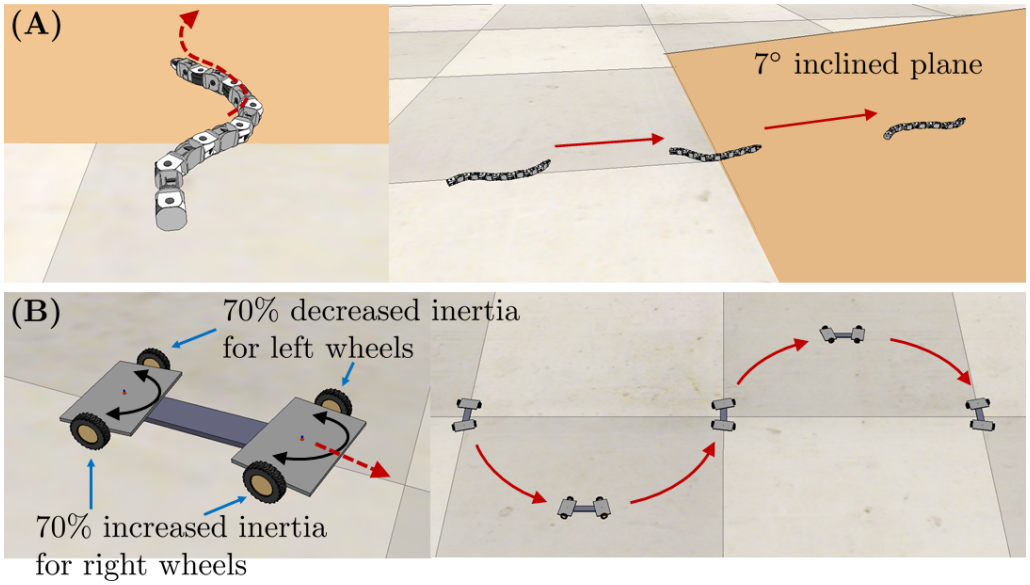

Figure 4-17: Snapshots of the simulation of snake-like robot crawling on an inclined plane (A) and pivotboard rotating on planes with different friction coefficients (B).

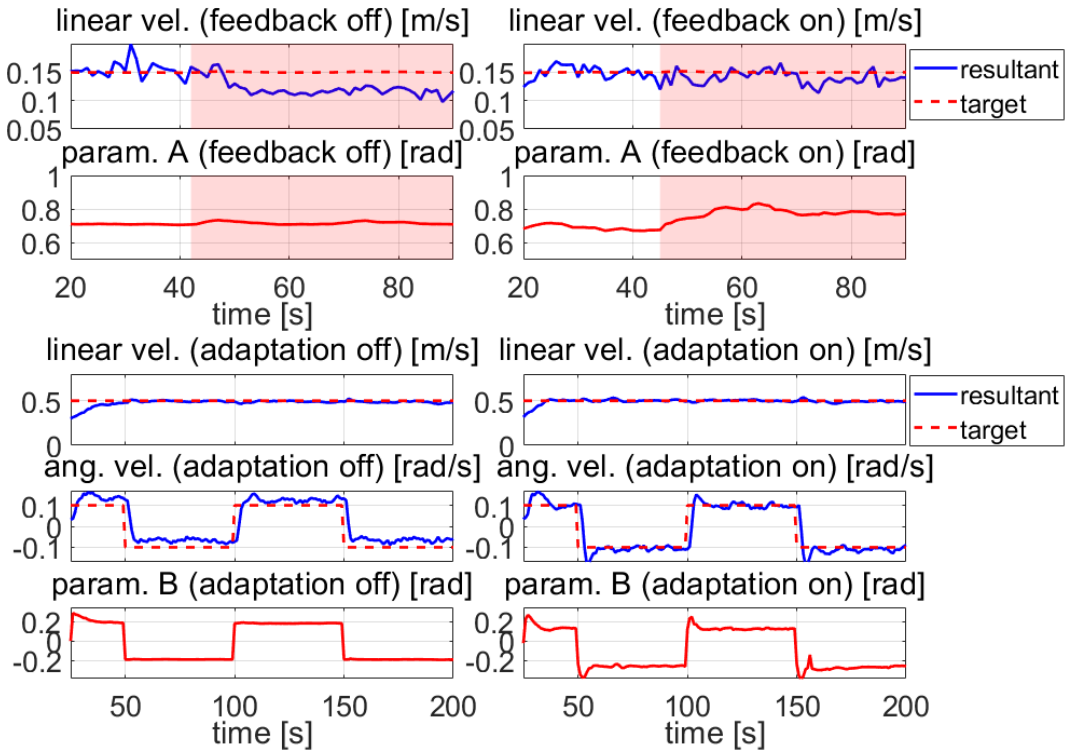

Figure 4-18: Simulation results of parameter adaptation with snake-like robot (top) and pivotboard (bottom).

the generated linear and angular velocity are 0.017 m/s and 0.063 rad/s. Similarly, simulations of desired acceleration and angular velocity generation are performed for CPG controlled pivotboard. For the simulation of desired acceleration, y (lateral) position is regulated to be zero, and the simulation result is shown in top of the Fig. 4-16, where the RMS error of the generated linear acceleration and y position are 0.013 m/s^2 and 0.071 m. Also for the simulation of desired angular velocity generation, linear velocity is maintained to be constant, and the simulation result is shown in bottom of the Fig. 4-16, where the RMS error of the generated linear and angular velocity are 0.018 m/s and 0.041 rad/s. Then, we also perform simulations to verify the CPG parameter adaptation law for two robotic systems. The snake-like robot is ordered to go through a 5° inclined plane as in the snapshot in (A) of Fig. 4-17, and the result in the top of Fig. 4-18 shows that with parameter adaptation, the generated velocity can be recovered to the desired value while being on the inclined plane. Also, the pivotboard is commanded to run and rotate on the planes with different friction coefficient (normal plane: 1.0, slippery plane: 0.3) as in (B) of Fig. 4-17, and the result is depicted in the bottom of Fig. 4-18 from which we can see that without the adaptation, the angular velocity is slightly below the desired value due to the slip, yet the angular velocity converges to desired value with the adaptation.

4.8 Conclusion

In this chapter, we present a new CPG-based motion generation framework to provide natural motion for f-LASDRA system and also for two other robotic systems operated with complex environmental interaction. For a natural motion generation, CPG models are formulated for each system based on phase oscillator mimicking the natural biomimetic or human generated motion. Especially for the f-LASDRA system, the CPG-generated motion is simulated with the simple water force model that brings a final target pose for each link of the system, which consequently makes

it act as if it is in water, which is an expected physical environment. We also present the acquisition of the inverse CPG model by machine learning for direct decision of CPG paramters that can generate desired motion. The proposed motion generation framework is then verified with simulations of three different robotic systems.

Chapter 5

Outdoor Flight Experiment of the F-LASDRA System

5.1 System Setup

To realize and verify the proposed CPG-based motion generation framework for the LASDRA system, we constructed a seven link LASDRA system as depicted in Fig. 5-1 so that much more natural motion can be realized with much increased degree of freedom (24 DOF). The design of link modules including actuators, joints are same with the system constructed in Sec. 3.5. For the onboard computation of semi-distributed state estimation algorithm and CPG based motion generation,

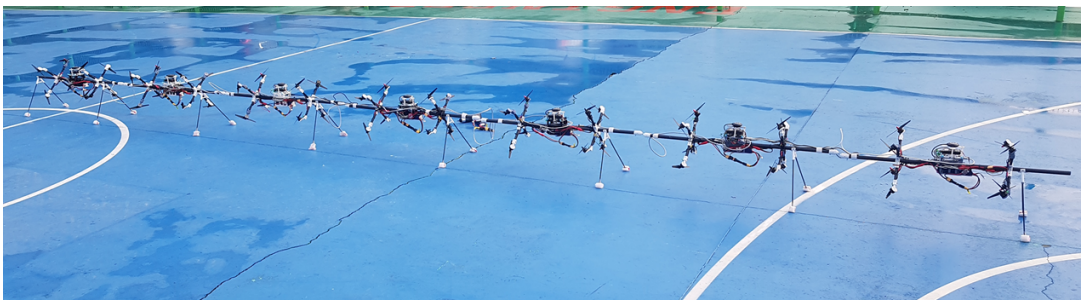

Figure 5-1: Seven link LASDRA system composed of Pixhawks on each link module and three Raspberry Pi's.

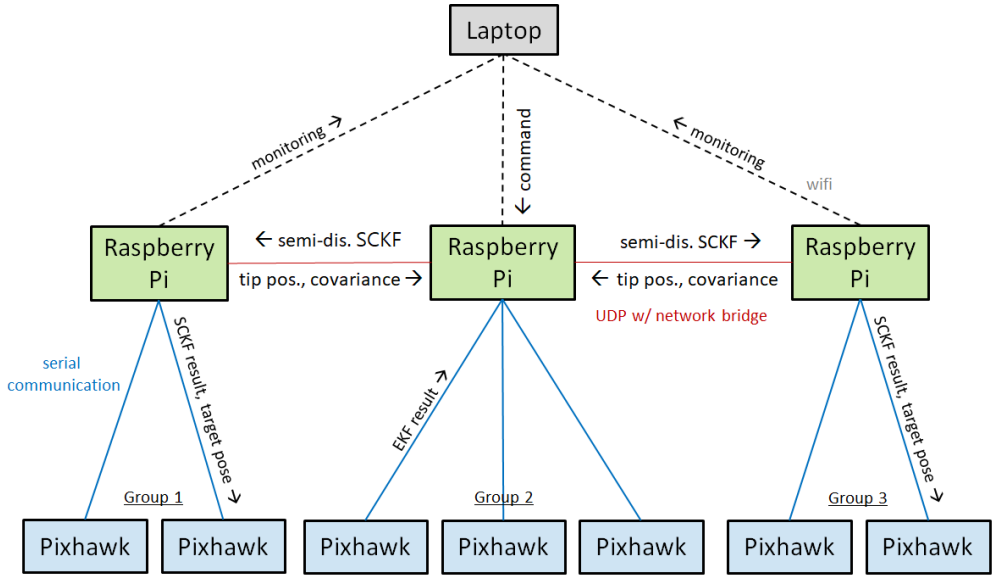

Figure 5-2: Diagram of command lines among computing modules exploited for seven link LASDRA system.

three Raspberry Pi's are used, each are controlling 2, 3, and 2 link modules. These onboard PCs send and receive required information each other through wired UDP communication with the rate of 50 Hz, and three PCs can be combined to a single network by exploiting network bridging. Among the onboard PCs, the central one takes a role of leader, conducting main computation of both semi-distributed estimation algorithm and CPG based motion generation by gathering information from the other PCs. The entire diagram depicting command line among the computing modules is show in Fig. 5-2.

5.2 Experiment Results

Using the flying LASDRA system constructed as in the previous section, we conducted several experiments in the outdoor environment to verify the CPG-based motion generation framework along with the state estimation algorithm both proposed in this thesis. First of all, an experiment of forward speed modulation is conducted through the CPG-based motion generation framework and the system

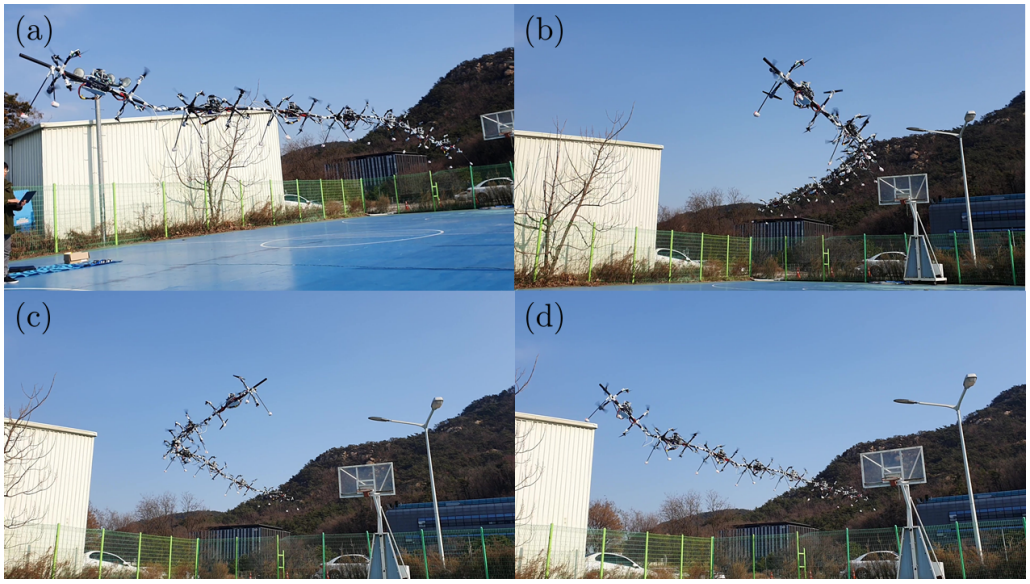

Figure 5-3: Snapshots of seven link LASDRA outdoor flying experiment with CPG-based motion generation.

is commanded to generate different forward velocities while undulating in horizontal direction. The result is shown in Fig. 5-4. The averaged values of RMS (root mean square) error magnitude of position and attitude with respect to the estimated pose were 15.79 cm and 7.74° respectively, and the plot at the bottom show that the entire body motion can track the target linear velocity using the inverse CPG model. The pose tracking error of each link, we believe, is mainly due to the low control gain, and the absence of the integral control by using decentralized impedance control. However, this performance is still tolerable by the compliant property of the controller. Also, the experiment of angular velocity generation is conducted for both yaw and pitch direction. The system is first made to follow different target yaw orientations with simple high-level proportional controller while undulating in vertical direction. The result is depicted in Fig. 5-5 where the average of RMS error magnitude of the seven links indicating the low-level pose tracking performance were 20.42 cm and 9.01°. Also, we can see that the desired linear and angular velocities are properly generated with the proposed CPG-based motion generation framework. Lastly, the experiment of gait transition is also conducted that the system is com-

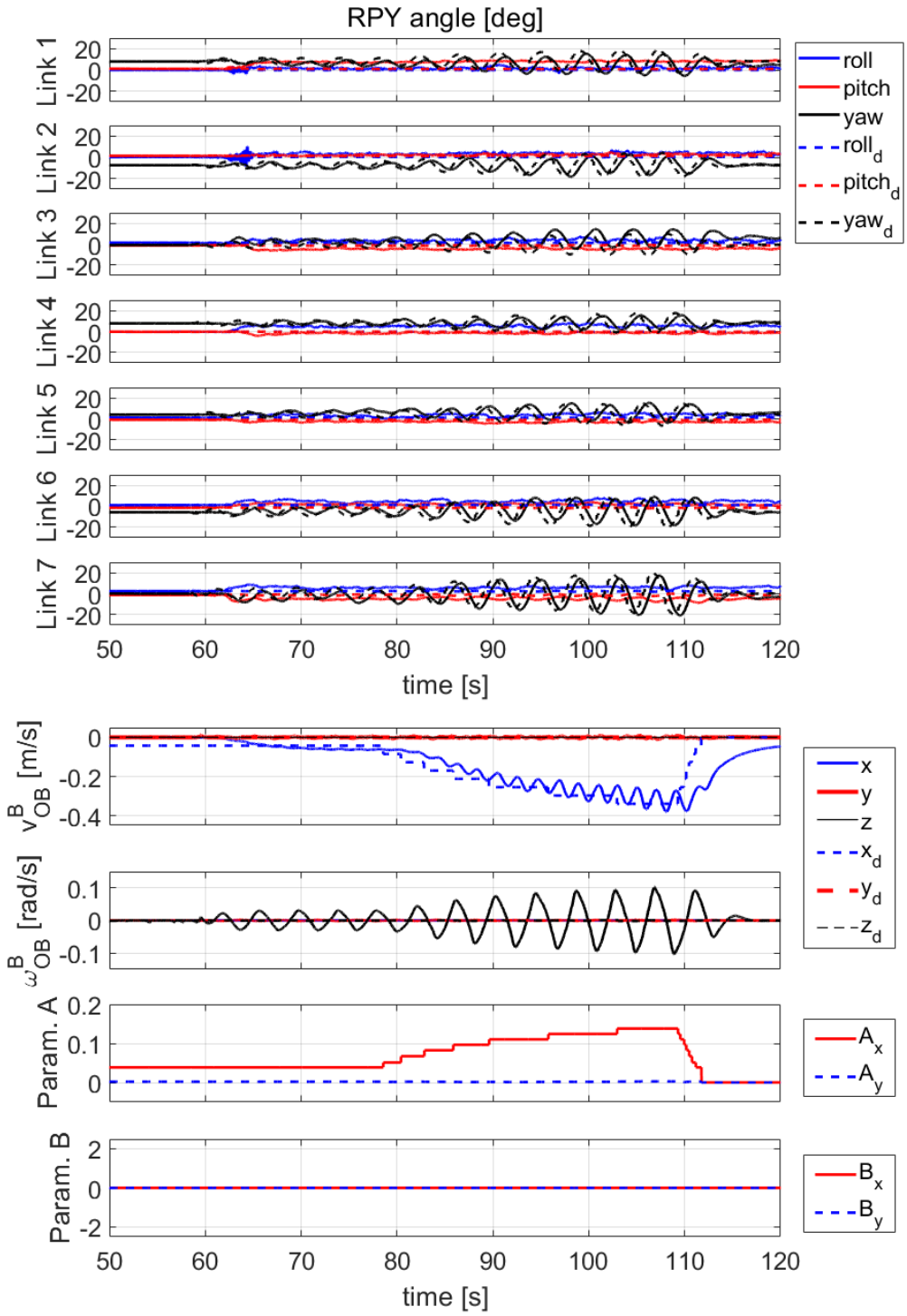

Figure 5-4: Forward speed modulation with CPG-based motion generation: (top) low-level attitude tracking performance at each link; (bottom) high-level CPG-based motion generation result.

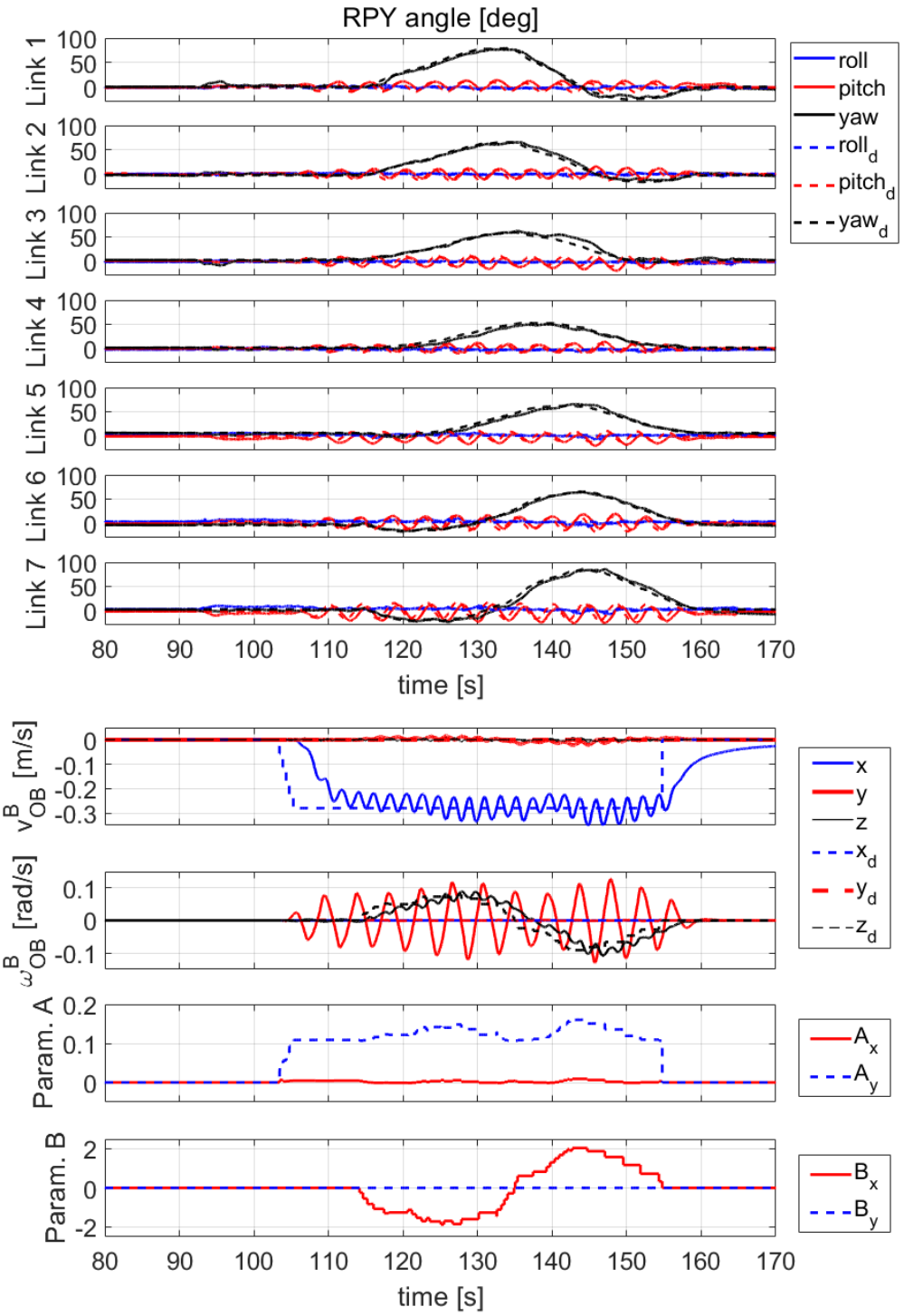

Figure 5-5: Yawing motion while undulating with vertical direction: (top) low-level attitude tracking performance at each link; (bottom) high-level CPG-based motion generation result.

manded to undulate alternately in horizontal and vertical direction. The experiment result is shown in Fig. 5-6 and we can see that the gait transition can be smoothly done by proposed motion generation framework. The averaged values of RMS error magnitude of pose tracking were 18.57 cm and 7.64° respectively.

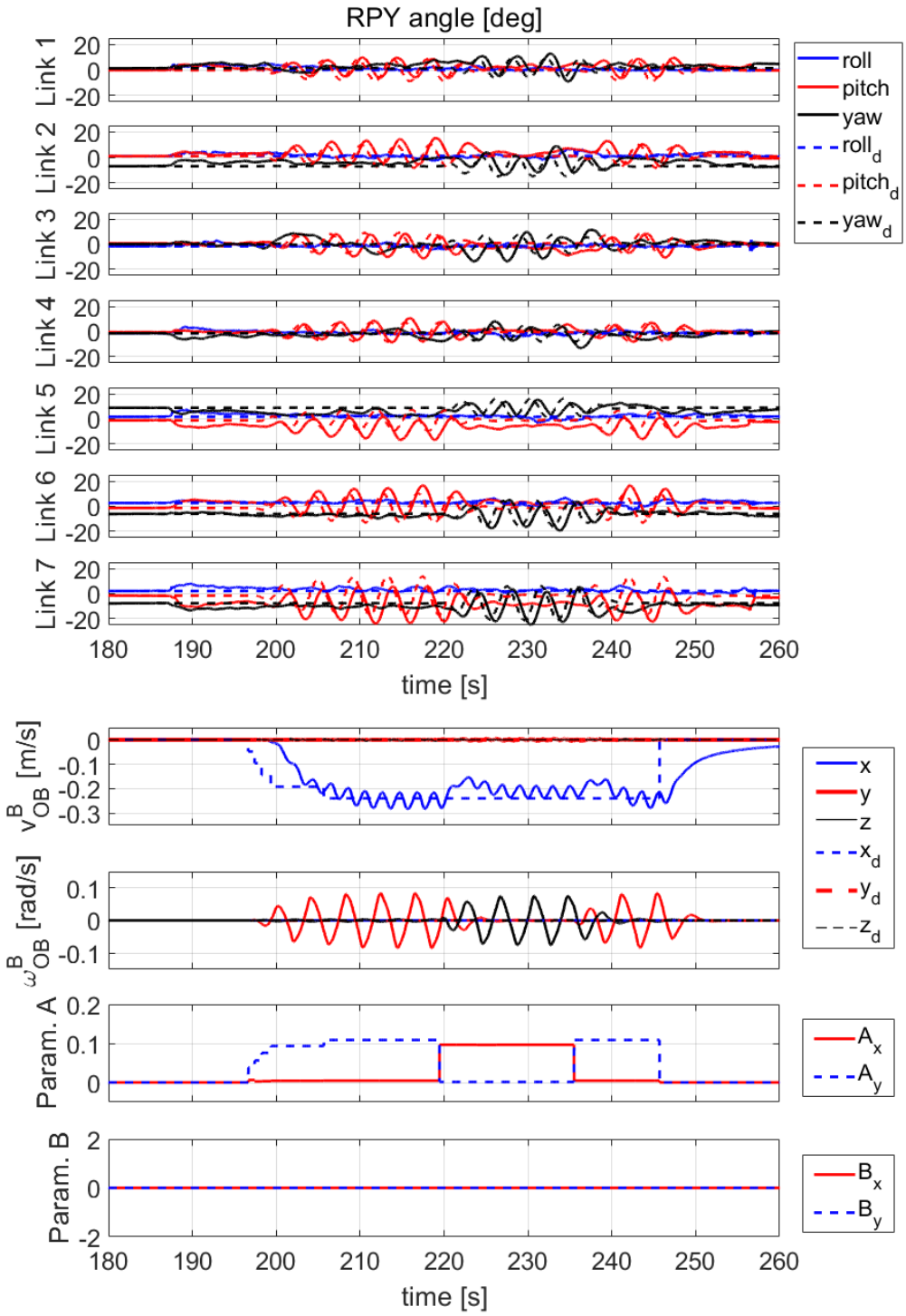

Figure 5-6: Gait transition between vertical undulation mode and horizontal undulation mode: (top) low-level attitude tracking performance at each link; (bottom) high-level CPG-based motion generation result.

Chapter 6

Conclusion

6.1 Summary

In this thesis, we present the contributions of developing key theoretical components for realizing a novel articulated aerial robotic system called f-LASDRA. We first develop an aerial robot called ODAR that can generate omni-directional wrench. To maximize the thrust margin of the ODAR system, we conduct design optimization that maximizes guaranteed minimum force and torque for any direction, while also considering some constraints including system volume, self weight compensation, and inter-rotor airflow avoidance to get a feasible solution. In addition, we propose a new control allocation method for the ODAR system to overcome the issue of delayed thrust output when changing the rotating direction (ESC-singularity) stemming from the sensorless motor and bi-directional thrust generation. Then, for the outdoor flight of the LASDRA system using IMU/GPS sensors on each link, we develop constrained Kalman filter based estimation framework along with its semi-distributed version of algorithm so that we can obtain pose estimation results satisfying kinematic constraint, while maintaining system scalability. Lastly, we propose a CPG-based motion generation framework that generates target pose of each link through a CPG model mimicking biomimetic motion and its simulation

with expected dynamic environment, and also incorporate inverse CPG model to enable direct generation of desired body motion while also maintaining the CPG generated natural motion.

6.2 Future Works

Some possible future research topics are as follows. Firstly, as the thrust is optimally allocated with infinity norm minimization in a single link ODAR system, we can also consider optimal wrench allocation for a group of links in LASDRA system by exploiting actuation redundancy (or physically appear as internal forces between the links). By doing so, a LASDRA link that demands more thrust than the other would be able to get help of neighbouring links.

Furthermore, teleoperation of the whole LASDRA system would be another interesting topic. Since we already have motion generation framework including CPG model, motion can be generated only using few parameters, and this can be a nice basis for a simple and intuitive teleoperation. Also, the information of the autoencoder output in the motion generation framework, meaning the nominal target motion, might be efficiently used for the user information, such as giving an information of difference between original and feasible target motion.

Bibliography

[1] Kuka. https://www.kuka.com/en-de/products/robot-systems/industrial-robots/kr-1000-titan. Accessed: 2019-11-21.

[2] Suidobashi heavy industry. http://suidobashijuko.jp/. Accessed: 2019-11-21.

[3] H. Yang, S. Park, J. Lee, J. Ahn, D. Son, and D. J. Lee. Large-size aerial skeleton system with distributed rotor actuation. In *Proc. IEEE Int'l Conference on Robotics & Automation*, pages 7017–7023, 2018.

[4] S. Park, Y. Lee, J. Heo, and D. J. Lee. Pose and posture estimation of aerial skeleton systems for outdoor flying. In *Proc. IEEE Int'l Conference on Robotics & Automation*, pages 704–710, 2019.

[5] G. Endo, T. Hagiwara, Y. Nakamura, H. Nabae, and K. Suzumori. A proposal of super long reach articulated manipulator with gravity compensation using thrusters. In *Proc. IEEE/ASME Int'l Conference on Advanced Intelligent Mechatronics*, pages 1414–1419, 2018.

[6] M. Zhao, K. Kawasaki, T. Anzai, X. Chen, S. Noda, F. Shi, K. Okada, and M. Inaba. Transformable multirotor with two-dimensional multilinks: Modeling, control, and whole-body aerial manipulation. *The International Journal of Robotics Research*, 37(9):1085–1112, 2018.

[7] T. Anzai, M Zhao, M. Murooka, F. Shi, K. Okada, and M. Inaba. Design, modeling and control of fully actuated 2d transformable aerial robot with 1 dof thrust vectorable link module. In *Proc. IEEE/RSJ Int'l Conf. on Intelligent Robots & Systems*, pages 2820–2826, 2019.

[8] M. Zhao, T. Anzai, F. Shi, X. Chen, K. Okada, and M. Inaba. Design, modeling, and control of an aerial robot dragon: A dual-rotor-embedded multilink robot with the ability of multi-degree-of-freedom aerial transformation. *IEEE Robotics & Automation Letters*, 3(2):1176–1183, 2018.

[9] M. Waibel, B. Keays, and F. Augugliaro. Drone shows: creative potential and best practices. Technical report, ETH Zurich, 2017.

[10] M. Ryll, H. H. Bülthoff, and P. R. Giordano. Modeling and control of a quadrotor uav with tilting propellers. In *Proc. IEEE Int'l Conference on Robotics & Automation*, pages 4606–4613, 2012.

[11] Y. Long and D. J. Cappelleri. Linear control design, allocation, and implementation for the omnicopter mav. In *Proc. IEEE Int'l Conference on Robotics & Automation*, pages 289–294, 2013.

[12] A. Oosedo et al. Flight control systems of a quad tilt rotor unmanned aerial vehicle for a large attitude change. In *Proc. IEEE Int'l Conference on Robotics & Automation*, pages 2326–2331, 2015.

[13] M. Ryll, D. Bicego, and A. Franchi. Modeling and control of fast-hex: a fully-actuated by synchronized-tilting hexarotor. In *Proc. IEEE/RSJ Int'l Conf. on Intelligent Robots & Systems*, pages 1689–1694, 2016.

[14] J. D. Geeter, H. V. Brussel, J. D. Schutter, and M. Decréton. A smoothly constrained kalman filter. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (10):1171–1177, 1997.

[15] M. A. Arbib and J. J. Bonaiuto. *From Neuron to Cognition via Computational Neuroscience*. MIT Press, 2016.

[16] G. Jiang and R. Voyles. A nonparallel hexrotor uav with faster response to disturbances for precision position keeping. In *Proc. IEEE Int'l Symposium on Safety, Security & Rescue Robotics*, pages 1–5, 2014.

[17] S. Rajappa, M. Ryll, H. H. Bülthoff, and A. Franchi. Modeling, control and design optimization for a fully-actuated hexarotor aerial vehicle with tilted propellers. In *Proc. IEEE Int'l Conference on Robotics & Automation*, pages 4006–4013, 2015.

[18] P. Roque and R. Ventura. Space cobot: modular design of an holonomic aerial robot for indoor microgravity environments. In *Proc. IEEE/RSJ Int'l Conf. on Intelligent Robots & Systems*, pages 4383–4390, 2016.

[19] A. Nikou, G. C. Gavridis, and K. J. Kyriakopoulos. Mechanical design, modelling and control of a novel aerial manipulator. In *Proc. IEEE Int'l Conference on Robotics & Automation*, pages 4698–4703, 2015.

[20] D. Brescianini and R. D'Andrea. Design, modeling and control of an omni-directional aerial vehicle. In *Proc. IEEE Int'l Conference on Robotics & Automation*, pages 3261–3266, 2016.

[21] A. E. Jimenez-Cano et al. Control of an aerial robot with multi-link arm for assembly tasks. In *Proc. IEEE Int'l Conference on Robotics & Automation*, pages 4916–4921, 2013.

[22] C. Korpela, M. Orsag, M. Pekala, and P. Oh. Dynamic stability of a mobile manipulating unmanned aerial vehicle. In *Proc. IEEE Int'l Conference on Robotics & Automation*, pages 4922–4927, 2013.

[23] H. Yang and D. J. Lee. Dynamics and control of quadrotor with robotic manipulator. In *Proc. IEEE Int'l Conference on Robotics & Automation*, pages 5544–5549, 2014.

[24] H. Yang and D. J. Lee. Hierarchical cooperative control framework of multiple quadrotor-manipulator systems. In *Proc. IEEE Int'l Conference on Robotics & Automation*, pages 4656–4662, 2015.

[25] S. Kim, S. Choi, and H. J. Kim. Aerial manipulation using a quadrotor with a two dof robotic arm. In *Proc. IEEE/RSJ Int'l Conf. on Intelligent Robots & Systems*, pages 4990–4995, 2013.

[26] S. Park, J. Lee, J. Ahn, M. Kim, J. Her, G-H. Yang, and D. J. Lee. Odar: aerial manipulation platform enabling omnidirectional wrench generation. *IEEE/ASME Transactions on Mechatronics*, 23(4):1907–1918, 2018.

[27] H-N. Nguyen, S. Park, J. Park, and D. J. Lee. A novel robotic platform for aerial manipulation using quadrotors as rotating thrust generators. *IEEE Transactions on Robotics*, 34(2):353–369, 2018.

[28] S. Park, J. Her, J. Kim, and D. J. Lee. Design, modeling and control of omnidirectional aerial robot. In *Proc. IEEE/RSJ Int'l Conf. on Intelligent Robots & Systems*, pages 1570–1575, 2016.

[29] P. Bosscher, A. T. Riechel, and I. Ebert-Uphoff. Wrench-feasible workspace generation for cable-driven robots. *IEEE Transactions on Robotics*, 22(5):890–902, 2006.

[30] R. M. Murray, Z. Li, and S. S. Sastry. *A mathematical introduction to robotic manipulation*. CRC, Boca Ranton, FL, 1993.

[31] F. Goodarzi, D. Lee, and T. Lee. Geometric nonlinear pid control of a quadrotor uav on se(3). In *Proc. IEEE European Control Conference*, pages 3845–3850, 2013.

[32] T. Tomić and S. Haddadin. A unified framework for external wrench estimation, interaction control and collision reflexes for flying robots. In *Proc. IEEE/RSJ Int'l Conf. on Intelligent Robots & Systems*, pages 4197–4204, 2014.

[33] G. Buondonno and A. De Luca. Combining real and virtual sensors for measuring interaction forces and moments acting on a robot. In *Proc. IEEE/RSJ Int'l Conf. on Intelligent Robots & Systems*, pages 794–800, 2016.

[34] D. J. Lee and K. Huang. Passive-set-position-modulation framework for interactive robotic systems. *IEEE Transactions on Robotics*, 26(2):354–369, 2010.

[35] D. J. Lee et al. Semiautonomous haptic teleoperation control architecture of multiple unmanned aerial vehicles. *IEEE/ASME Transactions on Mechatronics*, 18(4):1334–1345, 2013.

[36] Y. Zhang. Inverse-free computation for infinity-norm torque minimization of robot manipulators. *Mechatronics*, 16(3):177–184, 2006.

[37] R. Mahony, T. Hamel, and J-M. Pflimlin. Non-linear complementary filters on the special orthogonal group. *IEEE Transactions on Automatic Control*, 53(5):1203–1218, 2008.

[38] T. Tomić, C. Ott, and S. Haddadin. External wrench estimation, collision detection, and reflex reaction for flying robots. *IEEE Transactions on Robotics*, 33(6):1467–1482, 2017.

[39] Y. Lee, J. Yoon, H. Yang, C. Kim, and D. J. Lee. Camera-gps-imu sensor fusion for autonomous flying. In *Proc. Int'l Conference on Ubiquitous and Future Networks*, pages 85–88, 2016.

[40] L. Meier, P. Tanskanen, L. Heng, G. H. Lee, F. Fraundorfer, and M. Pollefeys. Pixhawk: A micro aerial vehicle design for autonomous flight using onboard computer vision. *Autonomous Robots*, 33(1-2):21–39, 2012.

[41] D. Simon. Kalman filtering with state constraints: a survey of linear and nonlinear algorithms. *IET Control Theory & Applications*, 4(8):1303–1318, 2010.

[42] J. Sola. Quaternion kinematics for the error-state kalman filter. *arXiv preprint arXiv:1711.02508*, 2017.

[43] D. Hauschildt, S. Kerner, S. Tasse, and O. Urbann. Multi body kalman filtering with articulation constraints for humanoid robot pose and motion estimation. In *Proc. Robot Soccer World Cup*, pages 415–426, 2011.

[44] G. Taga, Y. Yamaguchi, and H. Shimizu. Self-organized control of bipedal locomotion by neural oscillators in unpredictable environment. *Biological Cybernetics*, 65(3):147–159, 1991.

[45] N. Van der Noot, A. J. Ijspeert, and R. Ronsse. Bio-inspired controller achieving forward speed modulation with a 3d bipedal walker. *The International Journal of Robotics Research*, 37(1):168–196, 2018.

[46] H. Kimura, Y. Fukuoka, and A. H. Cohen. Adaptive dynamic walking of a quadruped robot on natural ground based on biological concepts. *The International Journal of Robotics Research*, 26(5):475–490, 2007.

[47] M. Ajallooeian, S. Gay, A. Tuleu, A. Spröwitz, and A. J. Ijspeert. Modular control of limit cycle locomotion over unperceived rough terrain. In *Proc. IEEE/RSJ Int'l Conf. on Intelligent Robots & Systems*, pages 3390–3397, 2013.

[48] A. J. Ijspeert, A. Crespi, D. Ryczko, and J-M. Cabelguen. From swimming to walking with a salamander robot driven by a spinal cord model. *Science*, 315(5817):1416–1420, 2007.

[49] T. Horvat, K. Melo, and A. J. Ijspeert. Spine controller for a sprawling posture robot. *IEEE Robotics & Automation Letters*, 2(2):1195–1202, 2017.

[50] J. Yu, M. Wang, M. Tan, and J. Zhang. Three-dimensional swimming. *IEEE Robotics & Automation Magazine*, 18(4):47–58, 2011.

[51] Y. Farzaneh and A. Akbarzadeh. A bio-inspired approach for online trajectory generation of industrial robots. *Adaptive Behavior*, 20(3):191–208, 2012.

[52] E. D. Tytell and G. V. Lauder. The hydrodynamics of eel swimming: I. wake structure. *Journal of Experimental Biology*, 207(11):1825–1841, 2004.

[53] A. P. Maertens, A. Gao, and M. S. Triantafyllou. Optimal undulatory swimming for a single fish-like body and for a pair of interacting swimmers. *Journal of Fluid Mechanics*, 813:301–345, 2017.

[54] J. A. Acebrón, L. L. Bonilla, C. J. P. Vicente, F. Ritort, and R. Spigler. The kuramoto model: A simple paradigm for synchronization phenomena. *Reviews of Modern Physics*, 77(1):137, 2005.

[55] S. Hirose and H. Yamada. Snake-like robots [tutorial]. *IEEE Robotics & Automation Magazine*, 16(1):88–98, 2009.

[56] J. Ostrowski, A. Lewis, R. Murray, and J. Burdick. Nonholonomic mechanics and locomotion: the snakeboard example. In *Proc. IEEE Int'l Conference on Robotics & Automation*, pages 2391–2397, 1994.

[57] E. Amrollah and P. Henaff. On the role of sensory feedbacks in rowat–selverston cpg to improve robot legged locomotion. *Frontiers in Neurorobotics*, 4:113, 2010.

[58] J. H. Barron-Zambrano, C. Torres-Huitzil, and B. Girau. Perception-driven adaptive cpg-based locomotion for hexapod robots. *Neurocomputing*, 170:63–78, 2015.

[59] J. Kim, J. Lee, and J. Lee. Central pattern generator parameter search for a biped walking robot using nonparametric estimation based particle swarm optimization. *International Journal of Control, Automation and Systems*, 7(3):447–457, 2009.

[60] A. Crespi and A. J. Ijspeert. Online optimization of swimming and crawling in an amphibious snake robot. *IEEE Transactions on Robotics*, 24(1):75–87, 2008.

[61] A. Spröwitz, R. Moeckel, J. Maye, and A. J. Ijspeert. Learning to move in modular robots using central pattern generators and online optimization. *The International Journal of Robotics Research*, 27(3-4):423–443, 2008.

[62] A. J. Ijspeert. Central pattern generators for locomotion control in animals and robots: a review. *Neural Networks*, 21(4):642–653, 2008.

[63] J. Yu, M. Tan, J. Chen, and J. Zhang. A survey on cpg-inspired control models and system implementation. *IEEE Transactions on Neural Networks and Learning Systems*, 25(3):441–456, 2013.

[64] Ö. Ekeberg. A combined neuronal and mechanical model of fish swimming. *Biological Cybernetics*, 69(5-6):363–374, 1993.

[65] M. Kim, Y. Lee, Y. Lee, and D. J. Lee. Haptic rendering and interactive simulation using passive midpoint integration. *The International Journal of Robotics Research*, 36(12):1341–1362, 2017.

[66] G. Gallego and A. Yezzi. A compact formula for the derivative of a 3-d rotation in exponential coordinates. *Journal of Mathematical Imaging and Vision*, 51(3):378–384, 2015.

[67] J. Gehring, Y. Miao, F. Metze, and A. Waibel. Extracting deep bottleneck features using stacked auto-encoders. In *IEEE Int'l Conference on Acoustics, Speech and Signal Processing*, pages 3377–3381, 2013.

초 록

분산된 로터로 구동되는 비행 스켈레톤 시스템의 디자인, 상태추정 및 제어

박 상 율
기계항공공학부
서울대학교

본 논문에서는 비행 스켈레톤 시스템 LASDRA (large-size aerial skeleton with distributed rotor actuation) 의 구현을 위해 요구되는 핵심 기법들을 제안하며, 이를 실제 LASDRA 시스템의 실외 비행을 통해 검증한다. 제안된 기법은 1) 전방향으로 힘과 토크를 낼 수 있고 충분한 가용 렌치공간을 가진 링크 모듈, 2) 높은 자유도의 다관절구조 시스템을 위한 위치 및 자세 추정 알고리즘, 3) 자연스러운 움직임을 내는 동시에 전체 시스템이 속도, 각속도 등 원하는 움직임을 내도록 할 수 있는 모션 생성 프레임워크로 구성된다.

본 논문에서는 우선 링크 모듈의 디자인을 위해 전방향으로 보장되는 힘과 토크의 크기를 최대화하는 구속 최적화를 사용하고, 실제 적용가능한 해를 얻기 위해 몇가지 구속조건(로터 간 공기 흐름 간섭의 회피 등)을 고려한다. 또한 센서가 없는 액츄에이터로 양방향 추력을 내는 것에서 야기되는 ESC 유발 특이점 (ESC-induced singularity) 이라는 문제를 처음으로 소개하고, 이를 해결하기 위해 선택적 맵핑 (selective mapping) 이라는 기법을 제시한다. 전체 LASDRA 시스템의 상태추정을 위해 시스템의 기구학적 구속조건을 만족하는 결과를 얻을 수 있도록 구속 칼만 필터 기반의 상태추정 기법을 제시하고, 시스템 확장성을 고려하여 반 분산 (semi-distributed) 개념의 알고리즘을 함께 제시한다. 마지막으로 본 논문에서는 자연스러운 움직임의 생성을 위하여 CPG 기반의 모션 생성 프레임워크를 제안하며, 기계 학습 방법을 통해 CPG 역연산 모델을 얻음으로써 전체 시스템이 원하는 움직임을 낼 수 있도록 한다.

주요어: 비행 스켈레톤, 디자인 최적화, ESC 유발 특이점, 확장성, 구속 칼만 필터, 중심 패턴 발생기, 기계 학습
학 번: 2013-20672